Predictive Models and Calibration Analysis in Large-Scale Computational Studies

by

Zhanyang Zhang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Statistics)
in The University of Michigan
2014

Doctoral Committee:

Professor Vijay Nair, Co-Chair
Professor Ji Zhu, Co-Chair
Professor Eunshin Byon
Professor Kerby Shedden

ACKNOWLEDGEMENTS

First, I would like to thank my advisors Professor Vijay Nair and Professor Ji Zhu, for their great guidance and wonderful support during my years in the University of Michigan. Also, Professor Kerby Shedden and Professor Eunshin Byon deserve my special thanks as my thesis committee members. Secondly, I would like to express my appreciation for all the help I received from the great people of the statistics department, as well as the CRASH team. At last, I would like to thank my parents, my wife Ting and my daughter Liz for their never-ending love and support.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

ABSTRACT

Predictive Models and Calibration Analysis in Large-Scale Computational Studies

by

Zhanyang Zhang

Chair: Vijay Nair and Ji Zhu

Computational modeling and simulation are used to study many complex phenomena where physical experiments are not feasible or too expensive. Examples include climate models, nuclear stockpile analysis, design and fabrication of integrated circuits, computer-aided manufacturing, and study of biological systems. Statistical methods play a variety of crucial roles in this area, ranging from the design of computer experiments to analysis of the outputs, developing predictive models, calibration analysis and, more generally, uncertainty quantification. This dissertation deals with two aspects of these statistical problems. The first part is concerned with developing statistical emulators for predictive modeling. In most applications of interest, a statistical model is used to fit the output from limited number of evaluations of the computational model, and the resulting "emulator" is used to approximate the input-output relationship. The current method of choice is a Gaussian Spatial Process (GaSP), where the output is viewed as the realization of a Gaussian process. While GaSP can be implemented using frequentist methods, it is most commonly used within a Bayesian framework. We compare the performance of GaSP with flexible regression-based approaches. These include existing methods such as multivariate adaptive regression splines (MARS), smoothing-spline anova (SS-ANOVA), multiple additive regression tree model (MART), and two methods developed in this dissertation: an expanded multivariate adaptive regression splines model (EMARS) and smoothing spline model with a kernel function based on exponential products (SS-Prod). Our empirical comparisons show that EMARS has better predictive performance than GaSP in a variety of situations. It is computationally much more efficient and it can be implemented using the current MARS algorithm. Given this computational advantage, it can be applied to more complex problems with many more input dimensions. The second part of thesis focuses on the calibration problem, where we have to determine the true (but unknown) values of certain input parameters to the computational model. This is a challenging inverse problem that suffers from identifiability issues. We develop conditions for determining identifiability and examine data-based approaches for checking the conditions in practice. The behavior of the methods is examined under various situations.

CHAPTER I

Introduction

Large-scale computational models are being increasingly used to study many complex phenomena in situations where physical experiments are infeasible or too expensive. This is typically the case in problems involving predictions such as climate modeling and weather forecasting. Another area where this approach has been used for a long-time is in examining nuclear stockpile readiness. Other examples include design of complex systems, such as aircraft, automobiles, transportation networks, and biological systems. In most of the applications, the output is multi-dimensional and is often a field. In practice, however, the scientists typically focus on one or a small number of output parameters (which may themselves be obtained by post-processing the output).

The study of this class of problems is characterized by a few distinct features. The computational models involve numerical solving partial differential equations, which are often computationally expensive. In some cases, even one evaluation of the code (one run) can take several days even on the largest and most efficient computing platforms. As a consequence, the number of runs (evaluations of the code as a function of the inputs) is limited. However, the input dimension can be large, which makes it challenging to study the input-output relationship. A common approach to this problem is to fit a statistical model to the observed input-output data and use the fitted model (called an emulator or sometimes a meta-model) for various inference problems. However, the model-fitting problem in this application is quite different from the usual ones with physical experiments because there is often very little “error” – the usual measurement error and other sources of random variation. In fact, in many situations, the input-output relationship can be viewed as deterministic in the sense that the same set of inputs will lead to the same output in the computational model. Thus, the major problem in developing a model is lack-of-fit rather than randomness. We will discuss the implications of this problem later in this section. In some of the applications, it is possible to conduct a (very) small number of field experiments which can be used to “validate” the computer runs and also do calibration analysis (to be discussed later).

However, many of the goals in the design and analysis of computational models are similar to what is done in physical experiments. These include: a) identifying the key input variables (also called variable screening in traditional design of experiments); b) understanding the nature of the influence of input parameters on the output (presence of nonlinear relationships and interactions as well as sensitivities); and c) developing good predictive models for the input-output relationships. They may also involve determining “optimum” values (maxima, minima, etc.). There are also some unique features. The types of experimental designs used are different since there is no (or very little) measurement error, so there is no need to replicate observations at the same input settings. The common class of experiments used are Latin hypercube designs and other types of space-filling designs (designs intended to cover the input space in some reasonable fashion) (see Santner et al. [61], Stein [68], Helton and Davis [33]). In the case of determining optimum values, sequential designs are often the most appropriate (see Robbins [58]). A second class of problems involves calibration where a subset of the input parameters involve unknown global constants whose values have to be determined from the combination of computer runs and field experiments. This is a challenging inverse problem as it involves unraveling a many-to-one relationship.

We use the application from the University of Michigan's Center for Radiative Shock Hydrodynamics (CRASH), of which we were members, to provide a concrete illustration. Radiative shocks exist in many applications in astrophysics including supernovae. The goal of the CRASH project was to study processes that simulate the radiative shock hydrodynamics in supernovae. To quote from the Center's website "In nature, radiative shock waves occur in supernovae, the most dramatic explosions in the universe. The shock waves that ripple from the demise of massive stars are so hot and fast that they emit radiation. These radiative shocks, in turn, change the structure and behavior of the exploding material, making the system difficult to simulate accurately with computers. That's why radiative shocks provide a great test case for research to improve predictive science." Research at the Center involved computational modeling of the shock waves as well as limited experiments at large laser facilities to create radiative shocks. The goal was to understand the difference between the simulation models and reality, quantify the uncertainty, and advance the "predictive science". CRASH was one of several centers that were supported under the Predictive Science Academic Alliance Program funded by the National Nuclear Security Administration (NNSA). The primary mission of NNSA is to "certify the safety of the US nuclear weapons stockpile."

Figure 1.1 provides a simplified version of the CRASH study. The X_H and D both denote input parameters and were used in a pre-processor (another computational model) to create Y_{HP} which were the input parameters to the CRASH simulator. For

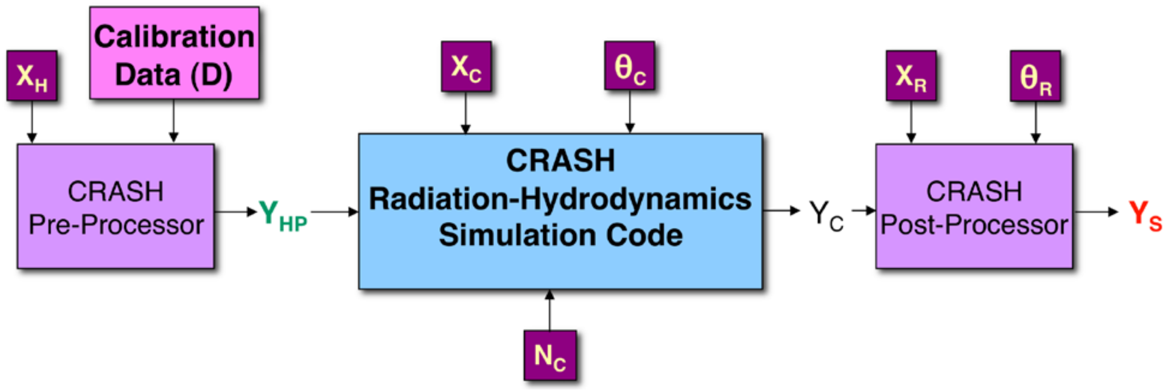


Figure 1.1: The input-output process of the CRASH code.

the purposes of this discussion, we can view X_H and D directly as input parameters. Examples of X_H included laser energy and pulse shape, initial Xenon pressure, Beryllium drive disk thickness, and geometry of the tube in which the shock wave is propagated. The goal was to understand the behavior of the shockwave propagation as these input parameters vary. D was composed of calibration parameters such as those involved in equation of state (EOS) and opacities. These are fixed but unknown constants, and the goal was to use the computational experiments and limited physical experiments to determine their values. There were also other “parameters” associated with the numerical aspects (such as the mesh parameters involved in the numerical solvers) that can contribute errors to the solutions, but we will not discuss them here. The actual output in this example was an image (x-ray radiograph) of the shock as it travels through the tube (see Figure 1.2). Certain features of this image were extracted and analyzed as finite-dimensional outputs. The primary ones were the shock positions at selected time points, the angle of the Xenon edge downstream from the shock, and the average thickness of the Xenon layer.

We partition the input parameters into $\{\mathbf{X}, \mathbf{\Theta}\}$ where $\{\mathbf{X}\} = \{\mathbf{x_1}, ..., \mathbf{x_p}\}$, the p-dimensional regular input parameters, and $\{\mathbf{\Theta}\} = \{\theta_\mathbf{1}, ...\theta_\mathbf{k}\}$, the k-dimensional

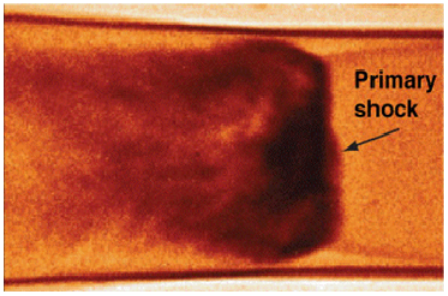


Figure 1.2: Primary shock in a xenon filled tube (physical experiment).

calibration parameters. Let Y denote the output. In the rest of this dissertation, we will assume Y is one-dimensional. One of the main goals is to approximate the input output relationship $f(\cdot)$ in

$$Y = f(\{\mathbf{X}, \mathbf{\Theta}\}). \tag{1.1}$$

As noted earlier, the input-output relationship $f(\cdot)$ in these computational studies is typically deterministic, i.e. the same input $\{\mathbf{X}, \mathbf{\Theta}\}$ yields the same output Y. Nevertheless, statistical approaches have been used to approximate the input-output relationship. Sacks and Welch (1989)[60] gave an early overview of the design and analysis of computer models. They discuss an approach based on Gaussian spatial processes (GaSP) where the output Y in $Y = f(\{\mathbf{X}, \mathbf{\Theta}\})$ is treated as a realization of a Gaussian spatial process on the $p+k-$dimensional input space. Various researchers have studied this formulation and constructed statistical emulators based on GaSP. In particular, the use of GaSP with a Bayesian approach has gained popularity over the past two decades. Additional formulations include

$$Y = f(\mathbf{X}, \mathbf{\Theta}) \ + \ \delta, \tag{1.2}$$

where δ is random (intended to capture additional sources of variation), may depend on the input parameters, and can itself be modeled by another Gaussian process. The popularity of the GaSP approach (with or without the Bayesian add on) can be attributed to the fact that inference about $Y = f(\{\mathbf{X}, \boldsymbol{\Theta}\})$ at the unobserved values of the input parameters becomes a prediction problem and there are standard approaches for quantifying the uncertainty. For simple cases, the prediction problem is straightforward since the corresponding conditional multivariate Gaussian distributions are easy to compute. There are also extensive results in the spatial analysis literature for more complex situations (various types of kriging). Bayesian inference when the parameters of the Gaussian covariance kernel are unknown is more complex, but recent developments in Markov Chain Monte Carlo methods can be used to compute the posterior distribution. Under the model in equation (1.1), with no error, the predictor will be an interpolator, i.e., it reproduce the observations at the observed points. Under equation (1.2), however, it will be a smoother in the sense that it will not reproduce the observations at the observed points. These are well-known points, and a recent review can be found in [61]. One major concern with the use of Bayesian GaSP, however, is computational complexity as this involved inverting a high-dimensional matrix multiple times. We will return to these issues in the next chapter.

The use of regression-based approaches to approximate the input-output relationship in equation (1.1) has been limited since they do not fit naturally into the framework where there is no random error or the random component is small in relation to the lack-of-model fit. Further, the uncertainty computations under the usual frequentist framework do not apply, and this has been viewed as a deficiency by practitioners. Again, we will return to this point in the next chapter. However, it still makes sense to consider regression-based approaches purely from an algorithmic point of view to develop predictors. There has been only one paper [4] in the literature that has compared the performance of regression-based approaches to the predictive models obtained from GaSP. The goal of Chapter 2 is to fill this gap by considering an extensive comparison of several regression-based approaches with Bayesian GaSP. Because the underlying input-output relationships are likely to be complex, we consider highly flexible regression-based methods. These include smoothing spline ANOVA (SS-ANOVA [29]), multivariate adaptive regression splines or MARS [19] and multivariate additive regression trees or MART[20]. The predictors from the regression-based approaches will not interpolate the observed data (as they implicitly assume the data are observed with error). Our main goal, however, is predictive performance (which will be formally defined in the next chapter). One advantage of these flexible regression models is the fact that the model building process usually involves minimizing the cross validated error as opposed to the training error, so that over fitting can usually be avoided. In the first part of the Chapter 2, we review four relevant methods –GaSP, MARS, MART, SS-ANOVA, and also proposed an expanded MARS model (EMARS) and a smoothing spline model with exponential product kernel (SS-Prod). Through numerical studies as well as real examples, we show that those alternative models can outperform the benchmark GaSP model. In particular, the EMARS approach does well under a variety of metrics.

Chapter 3 deals with calibration analysis. This problem usually arises in situation where we use both simulated and field data to infer some unknown parameters (parameters that are fixed and unknown constants). These calibration parameters are varied in the simulation or computational studies while they are fixed (by nature) at their true value in the field studies. So, intuitively speaking, the goal is to match the simulated data with the field data to determine the best match for the calibration parameters. This is a challenging inverse problem. Kennedy and O'Hagan [37] have proposed a Bayesian approach based on GaSP for calibration. The basic idea (expanded in more detail in Chapter 3) is to treat the simulated data and field data as the realizations of two correlated Gaussian Processes. The Bayesian approach combines both sources of data to do both prediction and calibration by accounting for multiple source of uncertainties. Higdon et al. [34] have extended the Bayesian calibration methods into cases with multiple outputs. Bayarri et al. [3] discussed calibration problems with functional outputs.

There are, however, some important issues related to calibration which have not been discussed much in the literature. The primary one deals with identifiability, and there are two kinds of identifiability. The first one is the presence of multiple solutions. It is intuitively clear that this can (and often will) happen since the the input-output relationship is not monotone. A more vexing problem is when the solution to the calibration problem lies in a lower-dimensional subspace. A simple toy example is where the computer simulator is $f(x_1, x_2, \theta_1, \theta_2) = x_1 + x_2 + \theta_1 - \theta_2$, and the field experiment comes from $y^f(z) = f(z_1, z_2, 0.2, 0.8) + \epsilon$. There is no way to distinguish the two calibration parameters θ_1 and θ_2 here. These issues are recognized among practitioners and researchers. In particular, there has been some discussion of the multiple solution problem; for example, authors who use Bayesian calibration techniques indicate that the posterior distributions will have multiple modes. Kennedy and O'Hagan note that the inference of calibration parameters is not necessarily related the "true" parameters, rather to determine the calibration parameters that "fit" the best for the purpose of physical process prediction. However, the identifiability issues have not been studied systematically. Chapter 3 develops a necessary and sufficient condition for identifiability and an additional sufficient condition. These are theoretical conditions that can be implemented only if the true function (input-output relationship) is known (not necessarily analytically but can be evaluated easily so that its derivatives can be computed). We then study some empirical methods for assessing these conditions based on the emulator. To address this issue, we propose a two-step solution – at first, estimate the unknown relationship between computer simulator output and its inputs using the computer data, for example using a Gaussian Process model or EMARS; then investigate the estimated function to see whether or not there is parameter redundancy in the calibration parameters. In this work, we first define different types of non-identifiability, and then developed several statistical methods to test the existence of such issues. Using simulations, we showed that our tests can be quite effective. For those identifiable calibration problems, we compared the widely used Bayesian GaSP calibration method and a proposed calibration approach based on EMARS. And our results show that the proposed method improves the GaSP model in both calibration parameter estimation and prediction of outputs.

CHAPTER II

Comparing different statistical approaches in predictive modeling of computational studies

This chapter deals with predictive modeling of the input-output relationship based on data from computer models. It provides a comparison of the performance of several approaches for prediction. Specifically, we compare the Gaussian Spatial Process (GaSP) approach with several techniques based on flexible regression modeling. These include smoothing spline ANOVA (SS-ANOVA), multivariate adaptive regression splines (MARS), and the multiple additive regression tree (MART). In addition, we also propose two modifications of existing methods, called expanded multivariate adaptive regression splines model (EMARS) and smoothing spline ANOVA with an exponential product kernel (SS-Prod). Our results showed that the EMARS method is a good competitor to GaSP. It can be implemented using existing MARS algorithms which makes it computationally faster, so we can apply it to situations with a larger number of input parameters.

2.1 Introduction

We provide an overview of the different approaches that will be considered in our study. As noted earlier, we will restrict attention to univariate outputs in this dissertation. In this chapter, we will not differentiate between regular input parameters and calibration parameters, and we will refer to the $p-$dimensional parameters as $\mathbf{x} = \{x_1, ..., x_p\}$. Suppose we execute the computer code at n input points. As noted in the last chapter, these are usually chosen according to some space-filling design. We let $(y_i, \mathbf{x}_i),\ i = 1, ..., n$ denote the "observations". Let $\mathbf{X}$ denote the $n \times p$ matrix of input values and Y denote the corresponding $n \times 1$ row vector of output values.

2.1.1 Gaussian spatial process (GaSP)

Sacks et al. [60] was among the first to describe a framework for inference that is based on treating the output as a realization of a Gaussian stochastic process (GaSP) $Y(x)$ with some mean function $\mu(x)$ and covariance matrix. (See Santner et al. [61] for a more recent discussion.) For example, the mean can be taken as

$$\mathrm{E}[Y(x)] = \sum_j \beta_j f_j(x)$$

with f_j being some known functions and β_j unknown parameters. There are several choices for the covariance functions [61, 56]. In this paper, we restrict attention to the popular product form:

$$\mathrm{Cov}(Y(x), Y(x')) = \frac{1}{\lambda} \prod_{j=1}^{p} e^{-\tau_j (x_j - x'_j)^q} \tag{2.1}$$

The parameter λ measures the overall precision; $\tau_j \geq 0$ measures the importance of one particular variable in the correlation; q controls the roughness of sample path. A typical choice of q is 2, in which the fitted function is very smooth.

The parameters β, τ, λ can be estimated using maximum likelihood, i.e., by maximizing the log-likelihood

$$l(\beta, \tau, \lambda) \propto$$

$$-\frac{1}{2}\log|\Sigma| - \frac{1}{2}(y - \sum_j \beta_j f_j(x))'\Sigma^{-1}(y - \sum_j \beta_j f_j(x))$$

where Σ is the covariance matrix generated using (2.1). With τ and λ known, the maximum likelihood estimate for β is given by

$$\hat{\beta} = (F'\Sigma^{-1}F)^{-1}F'\Sigma^{-1}Y$$

with F being the design matrix of $f_j(x_i)$. It should be noted that in practice, typically one assumes a simple mean structure, i.e., constant mean or zero mean (after first centering the response by subtracting its mean). There is no explicit solution for the MLEs of τ and λ, but they can be obtained numerically.

Prediction at a future input x^* can be obtained from the conditional distribution:

$$Y(x^*)|[Y_1, Y_2, ...Y_n] \sim N(\mathrm{E}[Y(x^*)|Y], \mathrm{var}[Y(x^*)|Y]).$$

Here

$$\mathrm{E}[Y(x^*)|Y] = \sum_j f_j(x^*)\hat{\beta}_j + \sum_{i=1}^{n} r(x_i, x^*)\hat{c}_i$$

where $r(x_i, x^*) = \mathrm{Cov}(Y(x_i), Y(x^*))$, and $\hat{c} = \Sigma^{-1}(Y - F\hat{\beta})$ and

$$\mathrm{var}[Y(x^*)|Y]) = \frac{1}{\lambda} - r'\Sigma^{-1}r$$

where $r = \{r(x_1, x^*), r(x_2, x^*).....r(x_n, x^*)\}$.

A Bayesian version of GaSP is often more commonly used [10, 34, 37, 52, 35]. This involves specifying prior distributions for the underlying parameters. For numerical reasons, it is common to assume an additional random error term to the GaSP, i.e $Y(x) = Z(x) + \epsilon$, where $Z(x)$ is a zero mean GaSP with spatial covariance defined by (2.1) and $\epsilon \sim N(0, \frac{1}{\lambda_e})$. Common choices of priors for those covariance parameters are:

$$\pi(\tau) \propto \prod_j (1 - e^{-\tau_j})^{-0.5} e^{-\tau_j}, \tau_j \geq 0$$

$$\pi(\lambda) \propto \lambda^{a-1} e^{-b\lambda}, \lambda > 0$$

$$\pi(\lambda_\epsilon) \propto \lambda_\epsilon^{a_\epsilon - 1} e^{-b_\epsilon \lambda_\epsilon}, \lambda_\epsilon > 0$$

with proper choice of $a, b, a_\epsilon, b_\epsilon$. The estimation of the posterior distribution of the parameters and the predictive distribution of the outputs are based Markov Chain Monte Carlo (MCMC) methods:

$$L(\tau, \lambda, \lambda_\epsilon | y_1, ...y_n) \propto \pi(\tau)\pi(\lambda)\pi(\lambda_\epsilon) \times L(y_1, y_2....y_n | \tau, \lambda, \lambda_\epsilon)$$

At a new point x^*, the inference of $Y(x^*)$ comes from the conditional distribution of

$$p(Y^* | y_1, y_2, ...y_n) = \int\limits_{\tau,\lambda,\lambda_\epsilon} p(Y^*|, y_1, y_2...y_n, \tau, \lambda, \lambda_\epsilon) p(\tau, \lambda, \lambda_\epsilon | y_1, ...y_n).$$

2.1.2 Smoothing Spline ANOVA and Smoothing Spline with Product Kernels

Smoothing splines have been discussed extensively in the literature [14, 63, 74, 29] although not very much in the context of computer models. To describe it, consider the regression problem $y_i = f(x_i) + \epsilon_i, i = 1, 2...n$, where $x_i \in [0, 1]$ and $\epsilon_i \sim N(0, \sigma^2)$. Common parametric methods, such as regression, assume that f is from a space spanned by known finite basis functions. The coefficients have to be estimated from data. For example, $f(x) = \sum_j \beta_j \phi_j(x)$, where ϕ_j are known basis functions. Smoothing splines (SS) allow f to be flexible enough to vary in a possibly infinite dimensional space. The underlying function space is spanned by a kernel function $R(s, t)$, which lies in a reproducing kernel Hilbert space (RKHS). The estimation of f is through the minimization of a penalized least square criterion,

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda\|f\|_{rkhs}^2$$

where $f \in H_0 \oplus H_1$ and $H_0 = \text{span}\{\phi_1, ...\phi_J\}$, $H_1 = \text{span}(R(s,t))$,with $R(s,t)$ the kernel function. It is known that the optimal estimator of $f(x)$ has an additive structure of parametric part and nonparametric part:

$$\hat{f}(x) = \sum_{j=1}^{J}\beta_j\phi_j(x) + \sum_{i=1}^{n}c_iR(x,x_i)$$

where β and c can be solved by minimizing the penalized least square criterion. The choice of λ, which controls how smooth the function is, turns out to be critical. Methods based on generalized cross validation (known as GCV) [11] can be used to estimate λ. When $\hat{\lambda} = 0$, the model will interpolate data; when $\hat{\lambda} \to \infty$, the model converges to least square estimate.

For computer models, generally speaking, the observational error is very small. Nevertheless, one ignores this and fits the (implicit) model $y_i = f(x_i) + \epsilon_i$. Thus a non-zero GCV estimate for λ does not interpolate the data.

One typical subclass of smoothing spline models is polynomial smoothing splines. (Without loss of generality, the support of the input variables can be restricted to $[0,1]$.) It seeks a minimizer of

$$\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda\int_0^1(f^{(m)})^2dx$$

in the space $C^{(m)}[0,1] = \{f : f^{(m)} \in L_2[0,1]\}$. This is a RKHS with $H_0 = \{f : f^{(m)} = 0\}$ and $H_1 = \{R(x,y)\}$, where one basis function of H_0 is polynomial functions

$\{x^k, k = 0, ...m-1\}$, and the reproducing kernel function is given by

$$R(x,y) = \int_0^1 \frac{(x-u)_+^{m-1}}{(m-1)!} \frac{(y-u)_+^{m-1}}{(m-1)!} du$$

It should be noted that there are other representation forms of the reproducing kernel functions, which leads to the same functional space. The above representation is one of the most commonly used forms.

In a p-dimensional problem, a weighted linear combination of the form

$$\tilde{R} = \sum_{l=1}^{L} \theta_l R_l$$

is often used, where $\{R_1, R_2 ....R_L\}$ corresponds to a tensor product set of one-dimensional kernels of degree d (see [29] for details). The total number of kernels involved L depends on both p and d. This is usually called Smoothing Spline ANOVA (SS-ANOVA), in which the model fits an ANOVA decomposition of the targeted function f using tensor-product kernels. Each of the parameters θ_ℓ (often called smoothing parameters) needs to be optimized from the data. When the dimension p is large, the number of smoothing parameters involved could become huge even with a mild choice of $d = 2$ (up to quadratic effects). As we will see, SS-ANOVA does not scale up to situations with even moderately large input space.

Therefore, we also consider an alternative: smoothing splines with product kernel (SS-Prod): Assume f is from a RKHS space associated with a kernel function $K(s,t)$,

$$K(s,t) = \prod_j e^{-\tau_j (s_j - t_j)^2}.$$

We estimate f by minimizing the criterion

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda\|f\|_{rkhs}^2.$$

Define $\theta = (\tau_1, ...\tau_p, \lambda)$, given θ, the explicit estimation of f can be given as discussed before. To estimate θ, we can use the generalized cross validation (GCV) criterion proposed by [74].

2.2 MARS, Expanded-MARS (EMARS) and MART

2.2.1 MARS and Expanded-MARS (EMARS)

Multivariate Adaptive Regression Splines (MARS) was first introduced by Friedman [19], and it has been applied in many applications. It is built on hinge functions of the form $[x_i - u]_+$ or $[u - x_i]_+$. Formally, MARS builds a model of the form

$$\hat{f}(x_1, ...x_p) = \alpha_0 + \sum_j \alpha_j B_j(x_1, ...x_p)$$

with each B_j being either a hinge function or product such as $[x_i - u]_+[x_j - v]_+[x_k - w]_+$. The knots are determined at the data values and the coefficients α are estimated using a least square form of criterion.

The model building process usually involves two stages: forward selection and backward pruning. At the first forward selection stage, the model starts with a constant term 1; then at each step, it finds a new pair of hinge functions $[x_i - u]_+$ and $[u - x_i]_+$ to make some new basis functions, which gives the maximum reduction in residual errors. Once a pair $([x_i - u]_+, [u - x_i]_+)$ got selected, it is removed from the set of candidate basic functions. Next, additional basis functions are added, based on the product of the existing set and all other available candidate basis functions.

For example, the next choice could be product of the form $[x_i - u]_+[x_j - v]_+$ or $[x_i - u]_+[v - x_j]_+$ where $i \neq j$.

The choice of which variable to use and the choice of knot u depends on a greedy search algorithm. The search can be done fairly fast by using a least-square update technique, and there are efficient algorithms in R. This process of repeatedly adding basis functions is stopped until the reduction in residual error is smaller than a preset threshold or some other condition.

To avoid over-fitting, a second stage – backward pruning is used in the algorithm. This deleting process is done one by one until some condition is satisfied. An existing term will be deleted if the resulting sub-model gives better cross validation error. After pruning, some unimportant variables will be eliminated from the model. The final model consists of a constant term, main effects $[x_i - u]_+$, two way interactions $[x_i - u]_+[x_j - v]_+$ and three-way interactions etc.

MARS has been shown not to do as well as GaSP for predictive modeling in comparative studies on computer models ([4]). It appears that the reason is because it explicitly does not allow interaction terms within the same variable. For instance, it would not allow terms like $[x_i - u]_+[x_i - v]_+$. To overcome this limitation, we considered an expansion of MARS by expanding the predictor space. [We will assume throughout the space of the input variables is non-negative. Typically we can scale the inputs to be in $[0, 1]$ before fitting emulators.] There are in fact two ways to do this.

1. include L copies of $(x_1, ...x_p)$ in the predictor space; or

2. in addition to $(x_1, ...x_p)$, add $\{(x_1^2, ...x_p^2); ....; (x_1^L, ...x_p^L)\}$.

To understand the difference between the two, consider just one input variable x and let v be a knot point. Further, suppose we take $L = 2$ so that we are only adding a quadratic term. Version1 adds the functions $[(x-v)_+]^2$ and $[(v-x)_+]^2$ as potential basis functions in addition to the first-order hinge function. Version two, however, will add the functions $[(x^2 - v^2)_+]$ $[(v^2 - x^2)_+]$ as new candidate functions. From a conceptual point of view, Version One seems more natural. However, in numerical comparisons, Version Two did slightly better, so we have restricted our analysis to this version.

So we have the following algorithm for Expanded MARS (EMARS):

Step 1: For a multi-dimensional problem with p predictors $(x_1, ...x_p)$, expand the predictor space from p to $L \times p$ by augmenting the original predictors with

$$\tilde{x} = (x_1, ...x_p; x_1^2, ...x_p^2; ....; x_1^L ....x_p^L)$$

Step 2: Fit the ordinary MARS model based on the expanded predictor space $\tilde{x}$.

The maximal augmenting parameter L and interaction degree d can be decided by minimizing the cross validated errors of the original MARS model. The idea is that we can use candidate values of L and d, then search for the optimal pair which has the smallest cross validated errors. In practice, we found that choosing d and L to be at most three (allowing up to cubic and third-order interactions) gives good results.

2.2.2 Multiple Additive Regression Trees

Multiple Additive Regression Trees (MART) was introduced by Friedman [20]. Like MARS, it is designed for approximating functions with multidimensional inputs. It uses regression trees as base functions (base learner).

Typically, given data $\{y_i; \mathbf{x_i}\}_{i=1}^n$, $\mathbf{x_i} = \{x_{1i}, x_{2i}...x_{pi}\}$, MART tries to approximate the function by using an additive structure

$$f(\mathbf{x}) = \sum_{j=0}^{M} \beta_j h(\mathbf{x}; \mathbf{a}_j)$$

where each $\mathbf{a}_j$ and β_j are parameters needed to be learned from data. The functions $h(\mathbf{x}; \mathbf{a})$ are regression trees indexed by parameter $\mathbf{a}$, i.e at each iteration step j, the p dimensional space is split into some disjoint hypercubes R_{lj}, and in each hypercube, a constant value is estimated from data:

$$h(\mathbf{x}; \mathbf{a}_j) = h(\mathbf{x}; \{R_{lj}\}_{l=1}^L) = \sum_{l=1}^{L} \bar{y}_{lj} 1(\mathbf{x} \in R_{lj})$$

The model fitting process involves M iterations. Starting with a constant term γ: $f_0(\mathbf{x}) = \text{arg min}_\gamma \sum_{i=1}^n \Psi(y_i; \gamma)$, where Ψ is the loss function, i.e a square loss. With $j >= 1$ till M, do the following:

1. Randomly take a subset with size $\tilde{n}$ from the training data, and $\tilde{n} < n$. Use the subset as training. (sample without replacement)

2. Calculate the pseudo residuals using gradient of the loss function. $\tilde{y} = -[\frac{\partial \Psi}{\partial f}]_{f=f_{j-1}(\mathbf{x})}$. With square error loss, the residual is $\tilde{y}_i = y_i - f_{j-1}(\mathbf{x}_i)$

3. Fit a L-terminal regression tree model on the residual data $\{\tilde{y}_i; \mathbf{x}_i\}$ to get the splitting rules $R_{lj}, l = 1...L$.

4. In each hypercube region R_{lj}, fit the coefficients of base learners:

$$\gamma_{lj} = \text{arg min}_\gamma \sum_{\mathbf{x}_i \in R_{lj}} \Psi(y_i, f_{j-1}(\mathbf{x}_i) + \gamma)$$

5. Update the current estimate by

$$f_j(\mathbf{x}) = f_{j-1}(\mathbf{x}) + \nu \cdot \gamma_{lj} 1(\mathbf{x} \in R_{lj})$$

where ν is the learning rate that need to be tuned when fitting the model and ν is typically set to be less than .1 to have better generalization error. Besides the learning rate ν, the number of trees M and tree size L also need to be tuned when fitting MART.

2.3 Numerical Studies

2.3.1 Study Design

In this section, we compare the performance of six statistical approaches on several different test cases.

- A: Additive function in five dimension:

  $$y = x_1 + 2x_2^2 - 0.5x_3^3 - 0.2\sin(x_4) + e^{-0.3x_5}$$

  where each x_i lies in $[-1, 1]$. This is a linear combination of very smooth functions. For training, we used a Latin hypercube design with 200 points on the 5-d space. To evaluate the predictive capabilities of the models, we used an independent Latin hypercube design with another 100 points.

- A': Five “inert” variables were added to case A:

  Namely, in addition to the five inputs in scenario A, we augment the input space with five inert (noise) variables $\{r_1, r_2, ..r_5\}$ that do not affect the response y. The goal is to test how each method performs in a high-dimensional input space with several unimportant input parameters. This is important in applications with factor sparsity where many inputs variables have little or no effect. It is of interest to examine how an emulator performs in such situations. We use the same set up for training and testing: 200 points (200 5-d LHS sample augmented with 200 5-d inert variables) as training and 100 LHS sample as testing.

- B: Five-dimensional function with additive structure plus two second-order interactions:

  $$y = \cos(x_1 + 2x_2) - x_3/(1 + x_4^2) + x_5^2$$

  where each x_i lies in $[-1, 1]$. This model is a bit more complicated than scenario A with the existence of low-order interaction. Similar to the settings in A, we choose 200 LHS sample for training and another 100 LHS points for evaluation purpose.

- B': Five inert variables added to the model in B:
  Similar to case A', we expand the five-dimensional input space of case B with additional five inert variables. Choose 200 LHS points as training and 100 LHS points as testing.

- C: 5-dimensional function with higher-order interactions:

  $$y = \frac{e^{-0.5x_1 - x2}}{1 + 0.2x_3^2 + 0.5x_4^2 + 0.6x_5^2}$$

  where each x_i lies in $[-1, 1]$. In this case, the response function is a non-linear function of all five input variables. Again, we choose 200 LHS as training and 100 LHS as testing.

- C': Five inert variables added to model in C.
  The analogous scenario of B' to B.

- D: Functions with highly local structures:

- D_1: We start with a one-dimensional function

  $$f(x) = [\sin(\pi x/5) + 1/5\cos(4\pi x/5)] * I(x < 10) + (x/10 - 0.8) * I(x >= 10)$$

  where $x \in [0, 20]$.

  The solid curve in the top left hand panel of Figure 2.1 shows the function. We see that it has a lot of local curvatures when $0 <= x <= 10$ and the function becomes linear when $x > 10$. Unlike the globally smooth functions in cases A, B, and C, this has many abrupt changes although the function is still continuous. We will return to a discussion of this figure and a comparison of the results. We selected 250 equally-spaced points on $[0, 20]$ for training, and an evaluation data set was chosen using 400 equally-spaced points on $[0, 20]$.

- D_2: We expand the function in D_1 to a 2-d function as follows:

  $$g(x_1, x_2) = f(x_1) * x_2/20$$

  where $x_1, x_2 \in [0, 20]^2$. Figure 2.2 shows the true function, and we can see the the interaction between x_1 and x_2 is localized. To assess the performance of the methods on this example, 200 LHS points on the 2-d space $[0, 20]^2$ was used for training, and a $20 \times 20 = 400$ full-grid design was used as the evaluation data set.

- D_3: Expand D_2 to five dimensions:

  Define

  $$y = f(x_1) * x_2/20 + (x_3/10 - 1)^2 * x_4/10 + cos(x_5/10)$$

  where each $x_i \in [0, 20]$. This is a combination of globally smooth functions with global interaction and locally smooth functions with local interactions. This is a very challenging scenario. For training, 500 LHS points were chosen, and 100 LHS points were chosen independently as the evaluation points.

  - D_3': Add 5 inert variables to the model in D_3:

    This setting is similar to those in A', B', C'.

- E: High-dimensional problem with factor sparsity:

  For this, we considered a 20-dimensional function with sparse input effects.

$$y = \frac{5x_{12}}{1+x_1} + 5(x_4 - x_{20})^2 + x_5 + 40x_{19}^3 - 5x_1 + 0.25x_{13}^2$$

$$+0.05x_2 + 0.08x_3 - 0.03x_6 + 0.07x_7 - 0.09x_9 - 0.01x_{10} - 0.07x_{11}$$

  with no effects on $(x_8, x_{14}, x_{15}, x_{16}, x_{17}, x_{18})$ (inert variables). The input domain is $[-0.5, 0.5]^{20}$. This is one of the testing functions used in Ben-Ari and Steinberg (1994) [4]. In the previous cases, the dimensions have been no larger than 10. Through this example, we want to test how those models perform on higher-dimensional problems with sparse inputs. Again, 200 LHS points were chosen as the training, and 100 LHS points were chosen independently as the evaluation points.

2.3.2 Results

We now examine the predictive performance of the six methods on the different test cases. Here prediction is based on RMSE (root-mean-squared-error) of the test data set. More formally,

$$RMSE = \sqrt{\frac{1}{n}\sum_{i}^{n}(y_i - \hat{y}_i)^2},$$

Table 2.1: Computational time comparison (elapsed CPU time in seconds)

Dimension	MARS	EMARS	SS ANOVA	MART	Bayesian GaSP	SS-Prod
5	0.067	0.199	11.731	13.742	15.871	14.574
10	0.089	0.641	NA	24.642	42.573	40.585
20	0.132	2.312	NA	58.57	142.313	138.542

where $\hat{y}_i$ is the predictor obtained from the test set and RMSE is computed over all the test samples. To get robust results, we replicated the results for each case (except for D_1). That is, we generated 20 LHS training and testing samples for each case and computed an $RMSE$ for each of the 20 replications. Table 2.2 shows the average $RMSE$ values. We will also examine the fitted functions and surfaces visually through the Figures 2.1 to 2.3 later. We also compared the computational times of model fitting using the 6 emulators on the same computer. Table 2.1 shows a comparison of the CPU time (in seconds) as the dimensions of the input space vary from 5 to 10 to 20. These computations were done on a laptop with Intel Core 2 Duo (2.26GHz) and 4GB memory. SS-ANOVA did not scale up to even 10 dimensions, While SS-Prod did, it is not competitive. B-GaSP is known to be computationally intensive as it is based on computing the inverse of high-dimensional covariance matrices. The performance of MART is in between but orders of magnitude higher than MARS and EMARS. The latter two are one to two orders of magnitude faster than the others. MARS is faster than EMARS since it deals with a lower-dimensional input space but, as we will see soon, its predictive performance is poor compared to the others.

Table 2.2 provides a comparison of the predictive performances based on $RMSE$. Here are main observations:

- In general, MARS does not perform as well as the other methods. EMARS outperforms MARS in all cases; while this is to be expected (since EMARS includes MARS as a special case), the extent to which it outperforms MARS

Table 2.2: Average RMSE on test data averaged over 20 replications. Top performers are indicated in bold.

Scenarios	MARS	EMARS	SS-ANOVA	MART	B-GaSP	SS-Prod
A - Additive	0.034	0.007	**0.002**	0.098	0.011	0.010
A' - Additive with inerts	0.038	**0.008**	NA	0.101	0.016	0.012
B - Lower order interaction	0.111	0.015	**0.007**	0.135	0.021	0.019
B' - B with inerts	0.121	**0.017**	NA	0.140	0.031	0.021
C - Higher order interaction	0.104	0.052	0.047	0.134	**0.022**	**0.021**
C' - C with inerts	0.124	0.055	NA	0.157	0.034	**0.023**
D_1 - Locally smooth 1-d	0.299	0.026	**0.008**	0.016	0.097	0.077
D_2 - Local interaction 2-d	0.098	**0.021**	0.029	**0.021**	0.057	0.055
D_3 - Combination on 5-d	0.117	**0.046**	0.069	**0.043**	0.082	0.077
D_3' - D_3 with inerts	0.119	**0.047**	NA	**0.049**	0.121	0.082
E - Higher dimension	0.021	**0.011**	NA	0.013	0.017	0.014

is quite surprising. Since the additional computational cost of EMARS is not that much more than MARS, one could conclude that MARS is inadequate and is dominated by EMARS for computational modeling.

- MART performs best when the input-output relationship is not as smooth, as is the case in the four sub-cases of D. This is to be expected based on our knowledge of regression trees. It is not competitive in the smooth cases of A, B, and C (and their higher dimensional versions A', B', and C').

- SS-ANOVA does extremely well in all applications with five or fewer input parameters. For the 5-dimensional problems, it performs the best in Cases A and B which are relatively smooth functions with global structures. D_2 and D_3 are two- and five-dimensional problems with more local structures, and SS-ANOVA does quite well (behind EMARS and MART). However, SS-ANOVA does not scale up to dimensions higher than five in the problems that we considered. The algorithms did not converge. Its computational cost is comparable to SS-Prod and B-GaSP.

- SS-Prod performs quite well overall. It is the best for situations C and C' that involve higher-order interactions. It dominates B-Gasp in almost all cases (except for a small difference for the case of C. Its computational cost is comparable to that of GaSP, which is considered to be computationally expensive. It does not do as well as EMARS, SS-ANOVA and MART for the four sub-cases on D with local features – the $RMSE$ are higher by a factor of two.

- B-GaSP, which is the method of choice in computational studies, performs reasonably well overall except for all the four sub-cases of D with a lot of local structure. As noted in the last item, SS-Prod outperforms it in terms of predictive performance in almost all cases.

- EMARS has the most "winners" among the cases studied: its performance is the best or very close to the best for A', B', D_2, D_3, D_4, and E. It is outperformed by SS-ANOVA in small dimensions: A, B, and D_1. It outperforms B-GaSP and SS-Prod in all cases except C and C' (cases with high-order interactions). The $RMSE$ is bigger by a factor of 2 in these two cases. In others, the RMSE's are smaller, and a lot smaller (up to a factor of 1/3) for the 4 sub-cases of D. EMARS also does well in high-dimensional cases with sparse inputs.

We now turn to a visual examination of some of the fitted input-output relationships and residuals. We consider first the performance of the six methods on the one-dimensional problem in Case $D1$, which has complex local features. As we can see from Figure 2.1, MARS performs quite poorly in this case. It ends up with three knots. The fit is piecewise linear as the original version of MARS does not allow for polynomials. On the other hand, EMARS does a good job of recovering the underlying function. It does not do as a good a job near the sharp curves, but its overall performance is quite good. B-GaSP does not perform as well in recovering some of the curvatures. The same is true for SS-Prod. On the other hand, SS-ANOVA and MART do extremely well, and this was seen in our discussion of the results in Table 2.2. Unfortunately, as discussed earlier, SS-ANOVA does not scale up to applications

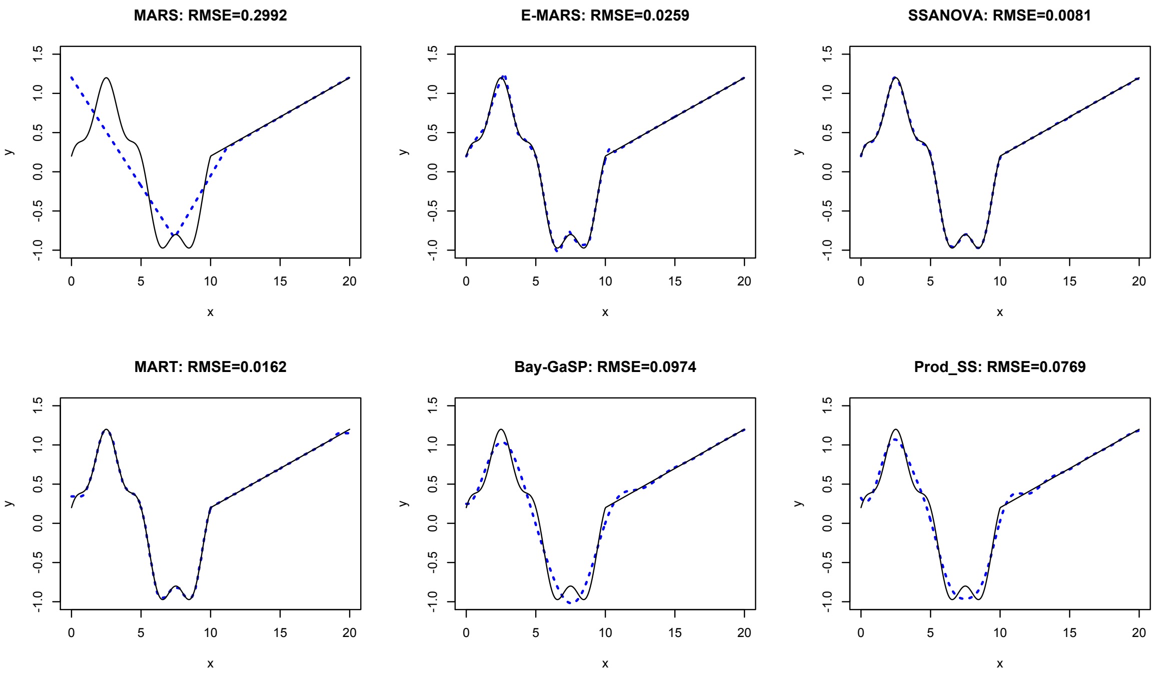


Figure 2.1: Case D_1 – A comparison of 6 emulators: MARS, EMARS, SS-ANOVA, MART, Bayesian-GaSP, SS-Prod: $f(x) = [\sin(\pi x/5) + 1/5\cos(4\pi x/5)] * I(x < 10) + (x/10 - 0.8) * I(x >= 10)$. The solid line shows the true function curve, while the dashed line shows the 6 fitted curves.

to dimensions much higher than five.

Figure 2.2, which corresponds to case D_2, is a two-dimensional version of $D1$ (Figure 2.1). Figure 2.3 shows the fitted response surfaces using the six different emulators. Overall, they all do a reasonable job of reconstructing the original surface except for MARS (top left hand panel). EMARS, SS-ANOVA, and MART do a better job than B-GaSP and SS-Prod in recreating the ridge in the middle (blue) as well as the shape of the flap (green) to the left.

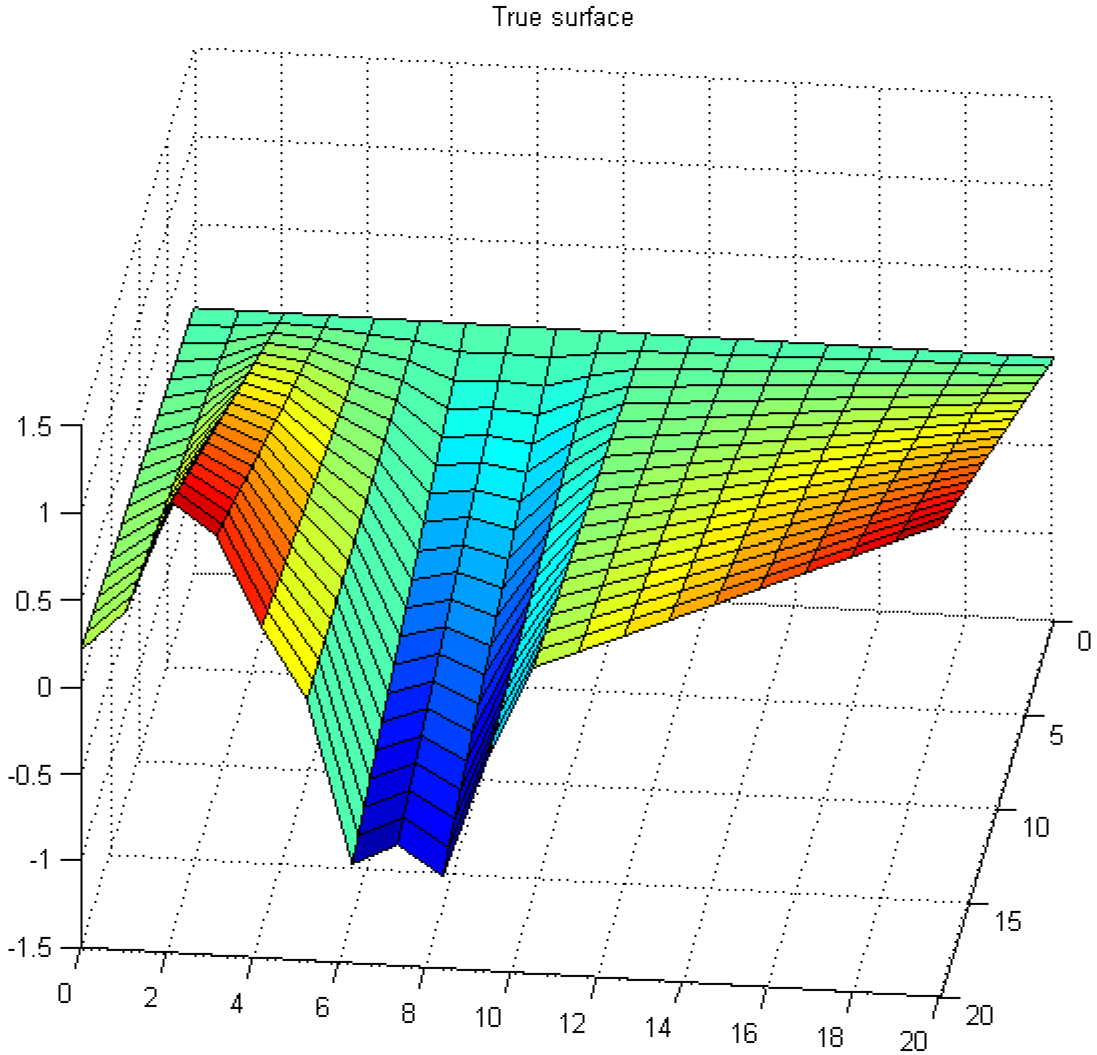


Figure 2.2: Case D_2 – True surface in scenario D_2: $g(x_1, x_2) = f(x_1) * x_2/20$, where $f(x) = [\sin(\pi x/5) + 1/5 \cos(4\pi x/5)] * I(x < 10) + (x/10 - 0.8) * I(x >= 10)$, and each $x_i \in [0, 20]$. The plot has been rotated so that a better visualization can be obtained.

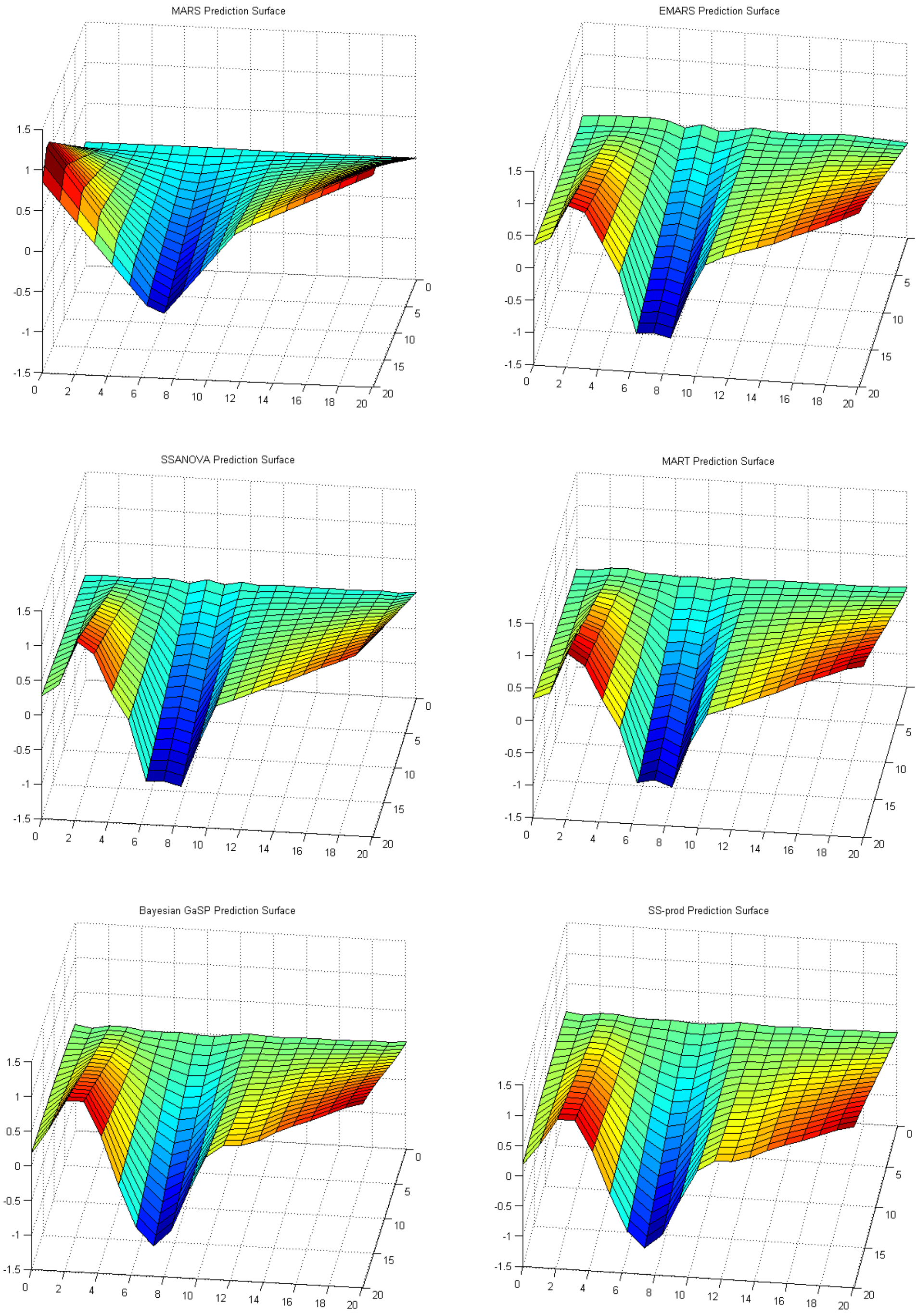


Figure 2.3: Case D_2 – Fitted surface on case D_2 using MARS (upper left), EMARS(upper right), SS-ANOVA(middle left), MART(middle right), Bayesian-GaSP (lower left), SS-Prod (lower right) respectively. The evaluation points are $20 \times 20 = 400$ points on the $[0, 20]^2$ space.

2.4 Assessing the Emulators on Illustrative Test Cases

We now use several illustrative test cases to compare the emulators. Some of these test cases have analytical expression for the input-output relationships and been used by other papers in the literature.

2.4.1 Piston simulator

This deals with an example for simulating a piston moving within a cylinder [4]. One of the quantities of interest is the time the piston takes to complete one cycle. The seven factors affecting the cylinder time are:

- M = Piston weight (kg), 30-60
- S = Piston surface area (m^2), $0.005 - 0.020$
- V_0 = Initial gas volume (m^3), $0.002 - 0.010$
- k = Spring coefficient (N/m), $1000 - 5000$
- P_0 = Atmospheric pressure (N/m^2), $9 \times 10^4 - 11 \times 10^4$
- T = Ambient temperature (K), 290 - 296
- T_0 = Filling gas temperature (K), 340-360

The input-output relationships has been modeled by the function:

$$\textbf{Cycle Time} = 2\pi\sqrt{\frac{M}{k + S^2 \frac{P_0 V_0}{T_0} \frac{T}{V^2}}}$$

where

$$V = \frac{S}{2k}\left(\sqrt{A^2 + 4k\frac{P_0 V_0}{T_0}T} - A\right)$$

and

$$A = P_0 S + 19.62M - \frac{kV_0}{S}.$$

Table 2.3: Comparison of RMSE on the Piston circular data

Data	MARS	EMARS	SSANOVA	MART	B-GaSP	SS-Prod
Piston simulator	0.028	**0.009**	0.013	0.089	0.012	0.011
Piston simulator with inerts	0.029	**0.014**	NA	0.091	0.025	0.016

Although there is an analytical expression for the input-output relationship, it is fairly complex.

To compare the methods, we chose the size of the training data to be 200, and the size of test size to be 100. Latin Hypercube designs were used to generate samples. We also added seven inert variables to test the performance under sparsity. Table 2.3 provides a comparison of the predictive performances of the six methods. EMARS does the best for both cases, with and without the inert variables, and its RMSE is almost a factor of 2 smaller than that of B-GaSP for the case with inert variables.

The fitted function using EMARS:

$$\hat{cycle.time} = 0.5177 + -67.2662(S - 0.0062)_+ + 72.3274(V0 - 0.0027)_+$$

$$+1813.7953(S^2 - 0)_+ - 33.2036(V0 - 0.0049)_+$$

$$-0.0025(V0 - 0.0027)_+(k - 1889.9493)_+ - 3209212.7568(V0 - 0.0027)_+(S^3 - 0)_+$$

$$-1.4965(V0 - 0.0027)_+(42.2078 - M)_+ + 1e - 04(M^3 - 81413.6129)_+(0.0099 - S)_+$$

$$+0.3477(S - 0.0113)_+(V0 - 0.0027)_+(k - 1889.9493)_+$$

$$-2.1987(V0 - 0.0027)_+(k - 1889.9493)_+(0.0113 - S)_+$$

The predicted model is not always the best for the purposes of interpretation. Nevertheless, examining this function provides an insight into the fitted model. We see that input factors P_0, T and T_0 do not appear in the model. This is confirmed from Figure 2.4, the factor-effects plot, which shows the relative contribution of the factors to the MSE. (To produce such plots, we vary one input across its domain and calculate the predicted output using EMARS, averaging over the other inputs.) S, piston surface area, has the major contribution, followed by V_0, initial gas volume. M and k have much smaller contributions. Returning to the fitted model from EMARS, we see that S has a cubic effect on the response. Figure 2.5 shows the one-dimensional marginal relationships between the inputs and output which demonstrate some of these conclusions through the projections to one-dimensional input-output relationships.

Figure 2.6 shows three-dimensional views of the relationship: Cycle time versus $(S,\ V_0)$. The left panel is the true marginal relationship between Cycle Time and $(S,\ V_0)$ (averaged over the other factors) and the right panel shows the fitted surface from EMARS. We can see the ridges in the fitted model. This is an artifact of using the hinge functions and is common to MARS-type fits. Figure 2.7 shows the one-way marginal residuals using EMARS for the six different methods. Again, we see the sharp ridges of MARS and to a lesser extent EMARS. Figure 2.8 shows the two-way marginal residuals. These figures provide additional (visual) comparisons of the overall performance of the six different emulators.

2.4.2 OTL Circuit

This application is based on codes to simulate an output transformerless (OTL) push-pull circuit [4]. The target variable of interest is the midpoint voltage (V_m) which is affected by the following six input variables.

- R_{b1} = Resistance $b1(K - Ohms)$, $50 - 150$
- R_{b2} = Resistance $b2(K - Ohms)$, $25 - 70$

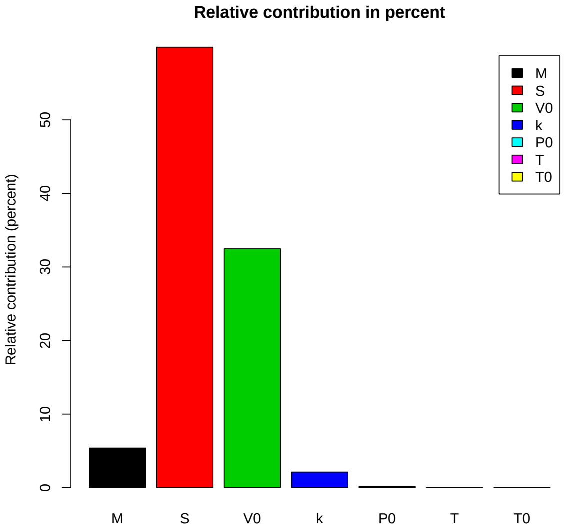


Figure 2.4: The relative variable contribution on the piston data analysis using EMARS.

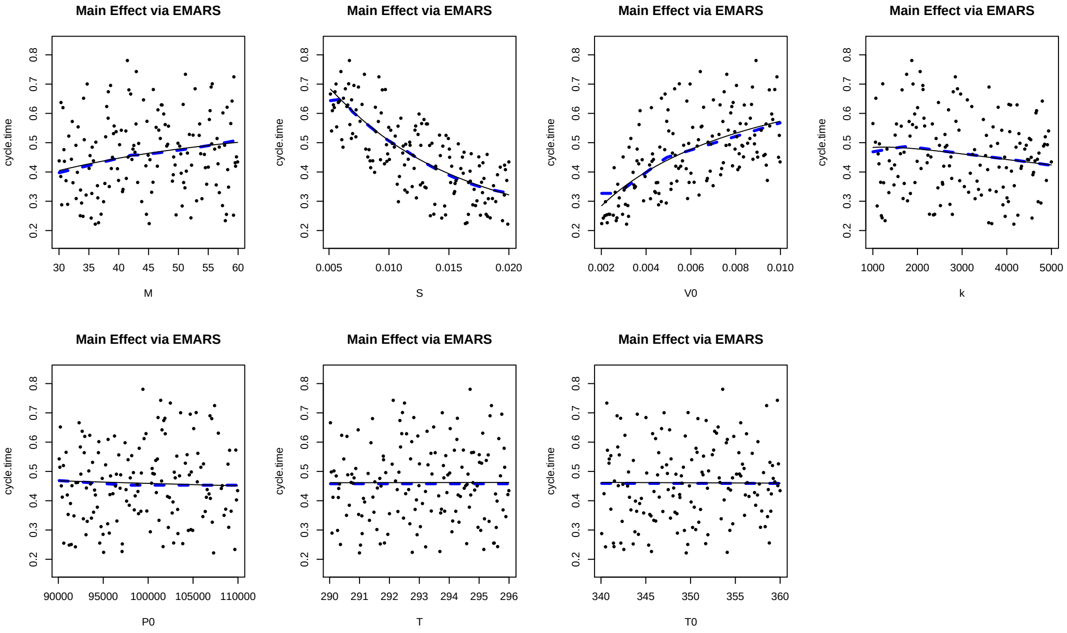


Figure 2.5: The one way marginal effects analysis on the piston data – true effects vs fitted effects using EMARS: solid black curves show the truth, while blue dashed lines show the fitted effects using EMARS. Black points are the observed points.

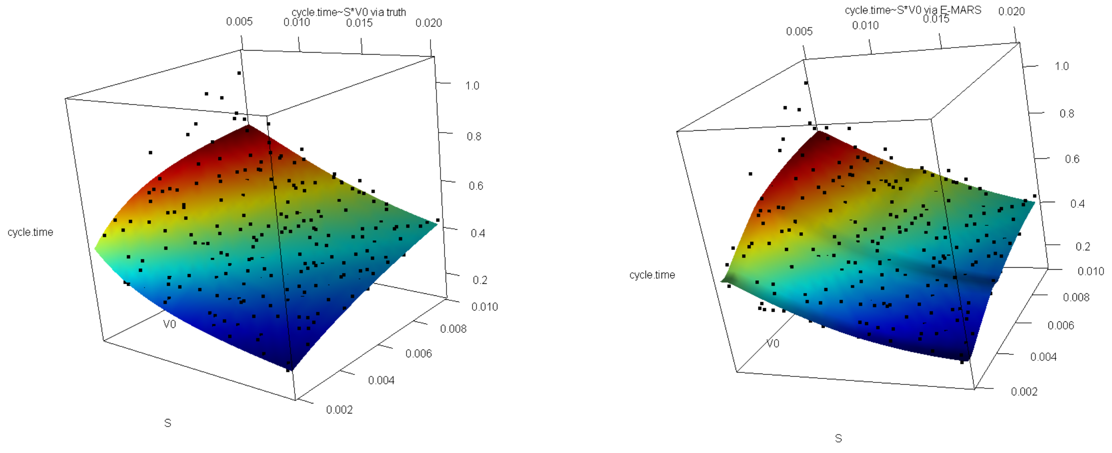


Figure 2.6: One example of a two way marginal effect analysis on the piston data – left panel shows the true marginal effects of *cycle.time* $\sim S \times V_0$; right panel show the fitted surface using EMARS.

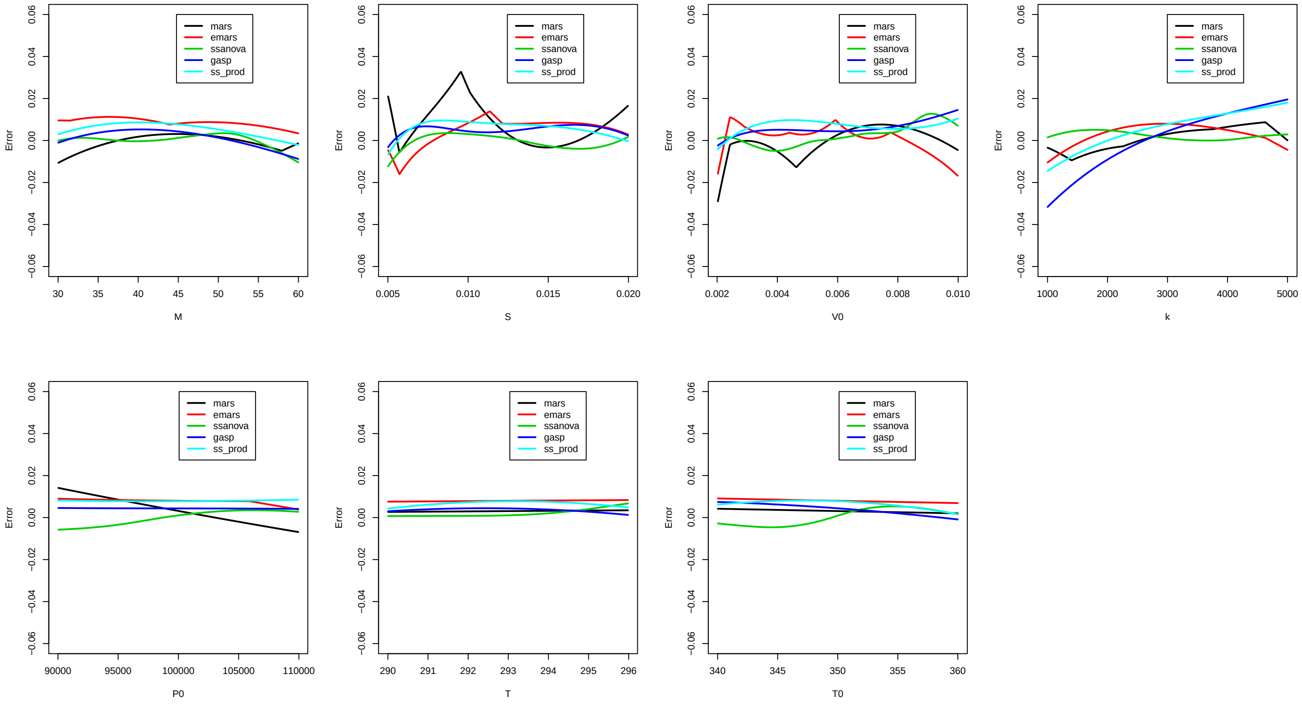


Figure 2.7: A comparison of one way marginal residual curves on the piston data analysis.

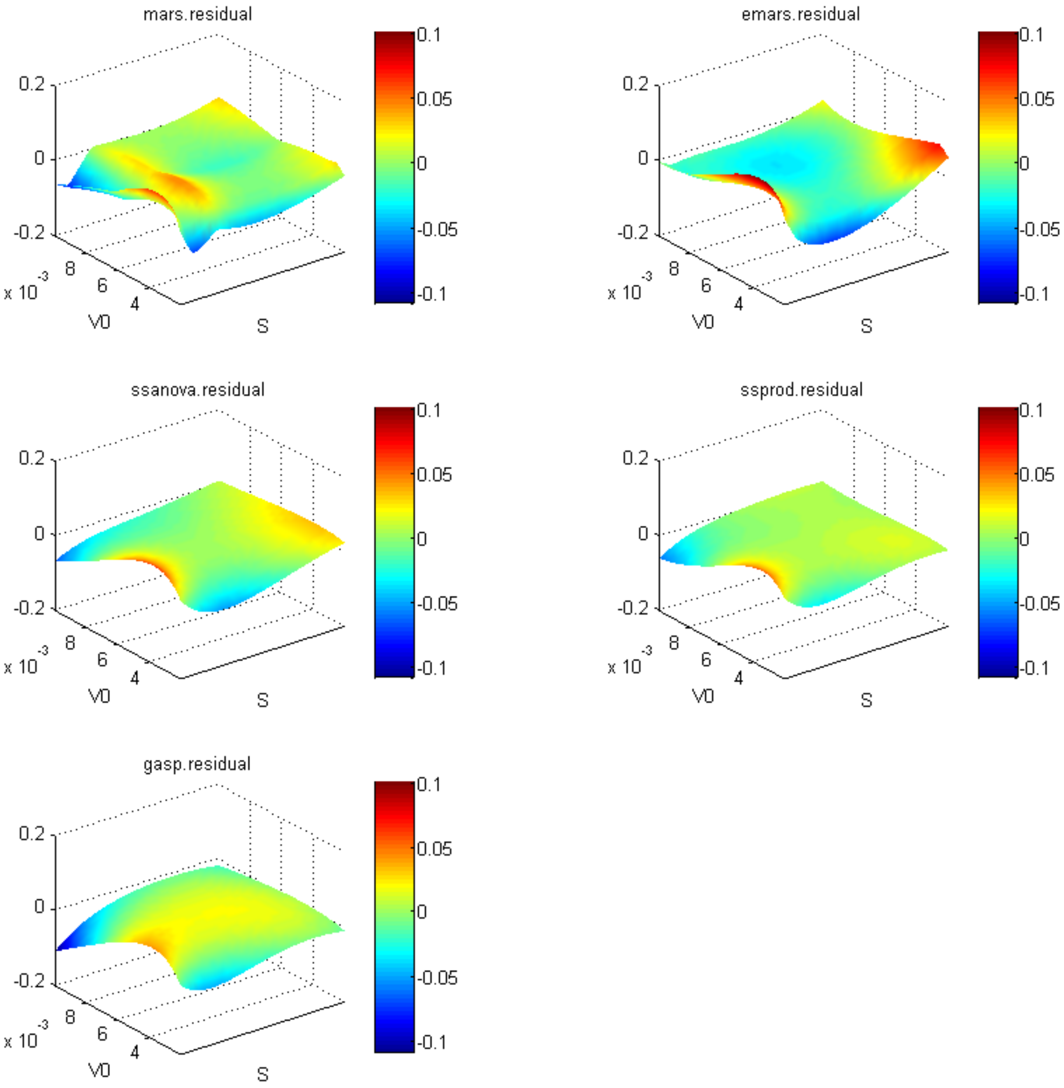


Figure 2.8: A comparison of two way marginal residual surfaces on the piston data analysis. The residual surfaces showed are: $(cycle.time - \widehat{cycle.time}) \sim S \times V_0$.

- R_f = Resistance $f(K - Ohms)$, $0.5 - 3$

- R_{c1} = Resistance $c1(K - Ohms)$, $1.2 - 2.5$

- R_{c2} = Resistance $c2(K - Ohms)$, $0.25 - 1.2$

- β = Current Gain (Amperes), $50 - 300$

$$V_m = \frac{(V_{b1} + 0.74)\beta(R_{c2} + 9)}{\beta(R_{c2} + 9) + R_f} + \frac{11.35R_f}{\beta(R_{c2} + 9) + R_f} + \frac{0.74R_f\beta(R_{c2} + 9)}{(\beta(R_{c2} + 9) + R_f)R_{c1}}$$

where

$$V_{b1} = \frac{12R_{b2}}{R_{b1} + R_{b2}}.$$

As before, we analyzed the application as is and with six inert variables. Again, we chose training size to be 200, and test size to be 100. Latin Hypercube designs were used to generate samples.

Table 2.4 shows a comparison of the predictive performance. Again, EMARS performed the best. The RMSE was about 1/2 of that of B-GaSP and SS-Prod for the situation with inert variables. The fitted function using EMARS is

$$\hat{Vm} = 4.769 + 1e - 04(Rb1^2 - 5682.2559)_+ + 1e - 05(5682.2559 - Rb1^2)_+$$

$$+0.1075(Rb2 - 60.1751)_+$$

$$-0.0943(60.1751 - Rb2)_+ + 0.3373(Rf - 2.5513)_+ - 0.3299(2.5513 - Rf)_+$$

$$-0.0359(Rc1^2 - 4.1695)_+ + 0.1835(4.1695 - Rc1^2)_+ - 0.0499(Rb1 - 111.7794)_+$$

$$+0.0519(111.7794 - Rb1)_+ - 6e - 04(Rb2^2 - 4205.3081)_+ + 4e - 04(4205.3081 - Rb2^2)_+$$

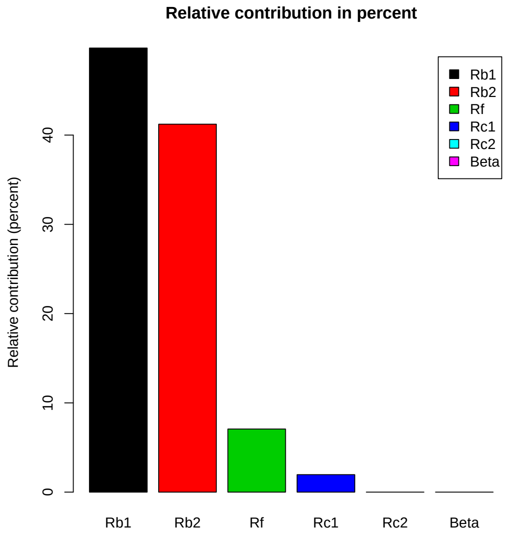


Figure 2.9: The relative variable contribution on the OTL data analysis using EMARS.

Table 2.4: Comparison of RMSE on the OTL Data

Data	MARS	EMARS	SSANOVA	MART	B-GaSP	SS-Prod
OTL circular	0.027	**0.007**	0.011	0.087	0.008	0.009
OTL circular with inerts	0.031	**0.007**	NA	0.088	0.015	0.011

$$+1e - 05(Rb1^3 - 1396641.9615)_+(60.1751 - Rb2)_+$$

$$-1e - 05(60.1751 - Rb2)_+(1396641.9615 - Rb1^3)_+$$

$$+0.109(Rf - 2.2089)_+(4.1695 - Rc1^2)_+ - 0.0877(2.2089 - Rf)_+(4.1695 - Rc1^2)_+ \ .$$

Only the factor β, current gain, does not appear in the fitted model. Although all the other five appear in the model, Figure 2.9 shows that $Rb1$ and $Rb2$ are the main contributors to variance explained. Figures 2.10, 2.11, 2.12, and 2.13 provide additional visual information on the fitted model.

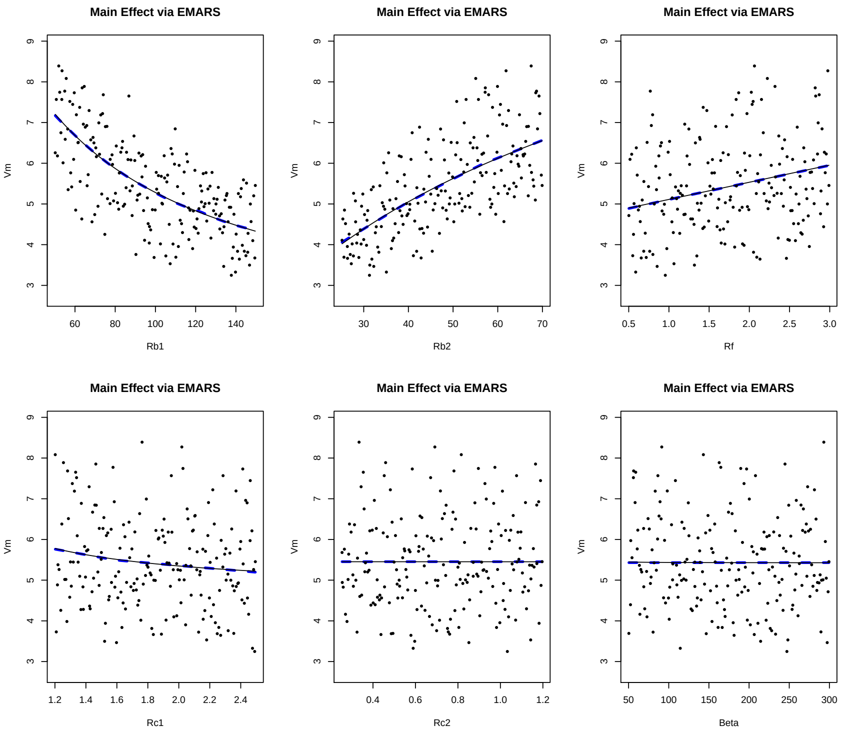


Figure 2.10: The one way marginal effects analysis on the OTL data – true effects vs fitted effects using EMARS: solid black curves show the truth, while blue dashed lines show the fitted effects using EMARS. Black points are the observed points.

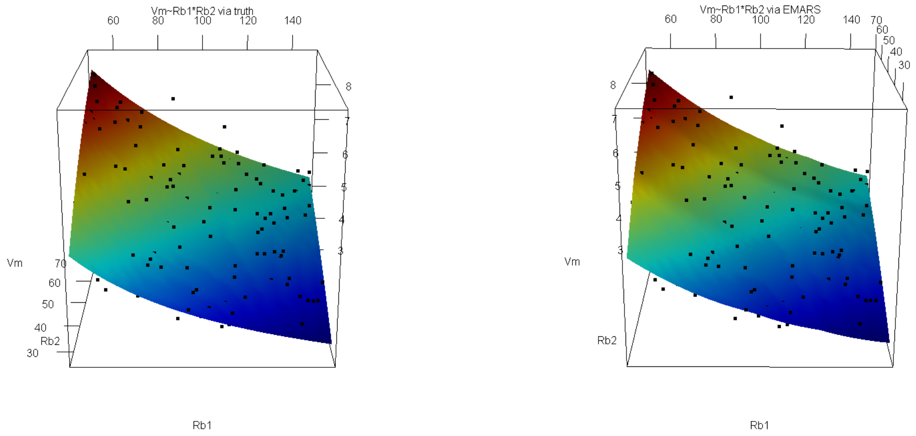


Figure 2.11: One example of a two way marginal effect analysis on the OTL data – left panel shows the true marginal effects of $V_m \sim Rb_1 \times Rb_2$; right panel show the fitted surface using EMARS.

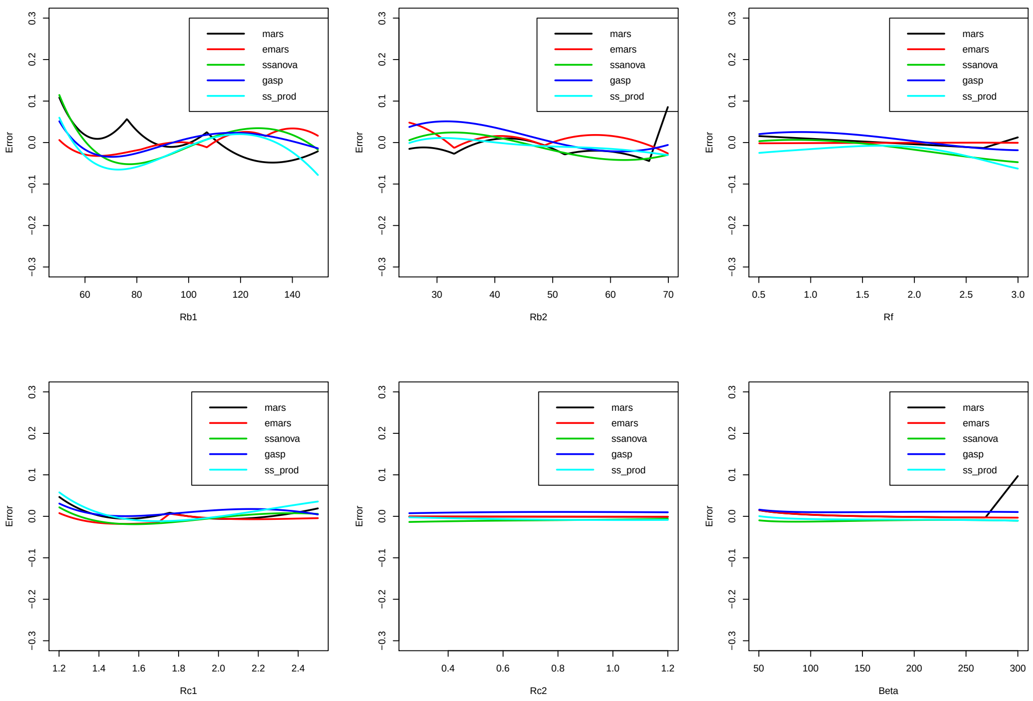


Figure 2.12: A comparison of one way marginal residual curves on the OTL data analysis.

2.4.3 The 512 Hyades Runs

This example is from the astrophysics applications at the University of Michigan Center for Radiative Shock Hydrodynamics (CRASH). As mentioned in the Introduction, the goal of this project is to develop and qualify the computational model for radiation hydrodynamics in applications that mimic supernovae [36]. As noted in this reference, the computational code simulates “an experiment where a laser is used to irradiate a Be disk and launches a radiative shock down a Xenon filled tube. Besides this primary shock traveling the tube, there is a second shock called “wall shock”, which is caused by the ablation of the tube wall because of radiation heat. The two shocks as well as the Xe-Be interface interacts together to produce a complex system. The physics involved is relevant to astrophysics and high-energy-density physics research. The CRASH code, which simulates this experiment, will help to gain insights into the physical process.”

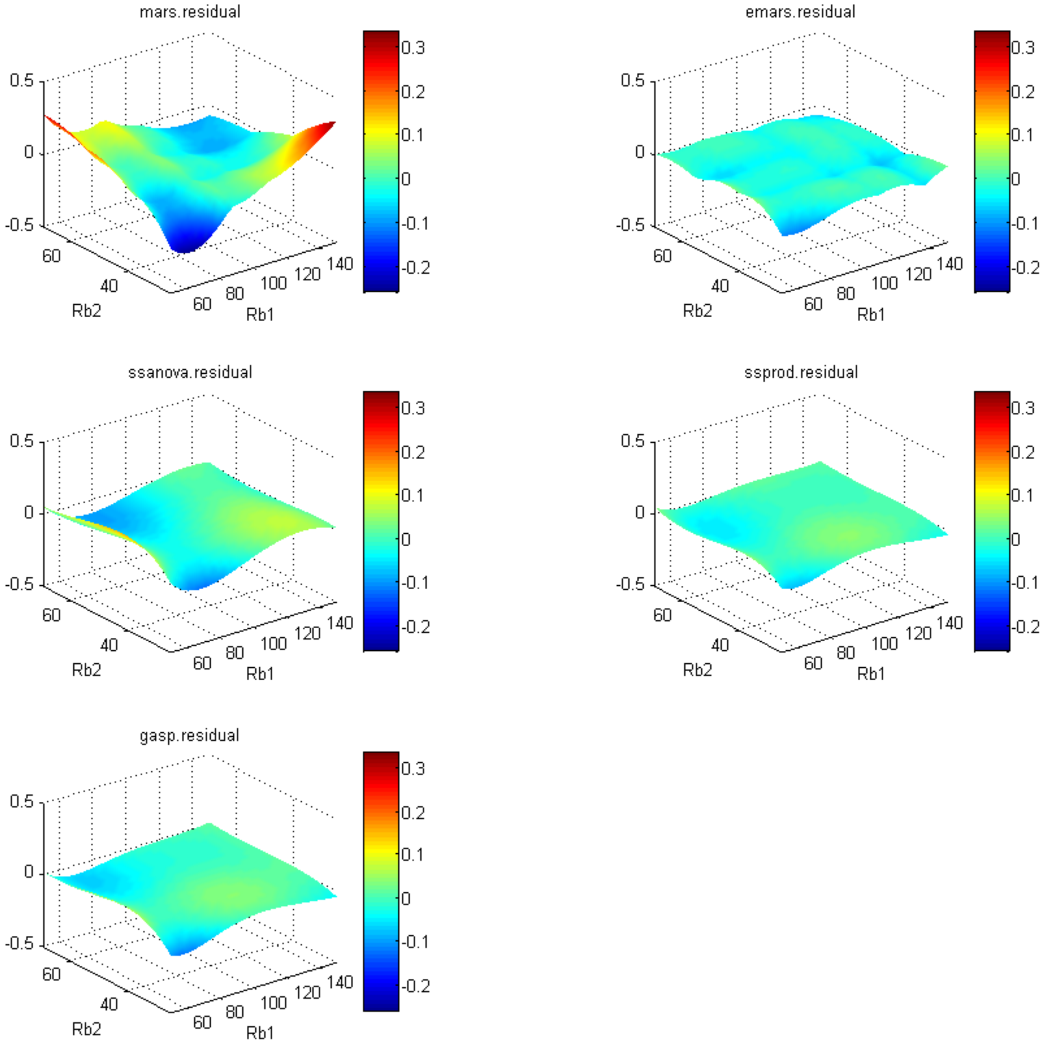


Figure 2.13: A comparison of two way marginal residual surfaces on the OTL data analysis. The residual surfaces showed are: $(V_m - \widehat{V_m}) \sim Rb_1 \times Rb_2$.

Table 2.5: Comparison of RMSE on the Hyades study

Data	MARS	EMARS	SSANOVA	MART	B-GaSP	SS-Prod
512 Hyades run	0.0030	**0.0017**	NA	0.0027	0.0032	0.0029

Because the CRASH code does not have the ability to model the first nanosecond of the experiment, the Hyades code [40] is used to develop initial inputs to the which will initialize the CRASH code. Our discussion here will focus on the Hyades code which has 15 input parameters and 40 output parameters. The 15 inputs include Be thickness, laser energy, xenon density, and other variables, including mesh parameters, Be opacity and Xe opacity parameters, etc. The 42 outputs measure a number of different quantities such as shock position, velocity, density and pressure. We pick one particular variable – the shock location, to test our emulators. A Latin hypercube design with sample size of 512 was used in the experiment. We hold out 100 of these points randomly for testing and use the rest to train the emulators.

Table 2.5 shows the predictive performance of the six methods on the Hyades example. Again, EMARS performed the best and does almost twice as well as B-GasP. Figure 2.14 shows the results of the analysis, using EMARS, for factor screening. There are 3 important variables with three others of lesser importance. Figures 2.15 and 2.16 show additional one- and two-way marginal plots from the fitted models.

2.4.4 Climate Applications

Finally, we turn to an application on climate models to compare the methods. The underlying computer model is NASA's Global Environmental Observing System (GEOS-5) (see Suarez et al. [70]), which is used to study the hydrological cycle for oceanic and land-based deep convection. In this application, there are 19 inputs, describing variety of cloud, convection parameters used in the model. The output variable used in this study describes the convective precipitation rate (in mm/hr) in

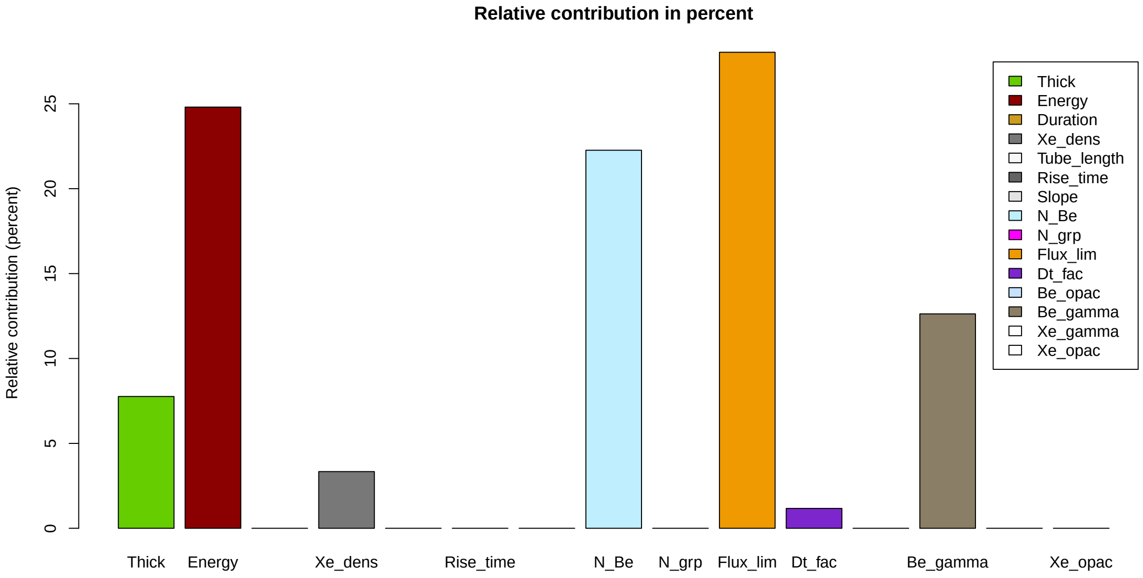


Figure 2.14: Relative variable contribution of the 15 inputs in the Hyades case using EMARS model.

Table 2.6: Comparison of RMSE on the climate data study

Data	MARS	EMARS	SSANOVA	MART	B-GaSP	SS-Prod
Climate data	0.0092	**0.0021**	NA	0.0081	0.0052	0.0031

a certain region. In additional to those 19 inputs, we have one "inert" input to test emulators' variable screening abilities. The number of design points are 452, which covers the 20-dimensional input space using a Latin Hypercube design. We fitted the six methods using 400 randomly chosen data as training sample, and 52 points as the independent validation set. Figure 2.17-2.20 shows the variable contribution as well as the marginal sensitivity plots. The prediction metrics based on RMSE on validation set shows EMARS stands out (see table 2.6). The results are discussed further below.

2.4.5 Summary of the Comparisons

On the piston simulator study, the four methods EMARS, SSANOVA, Bayesian GaSP and SS-Prod all did quite well, with EMARS having some edge over others.

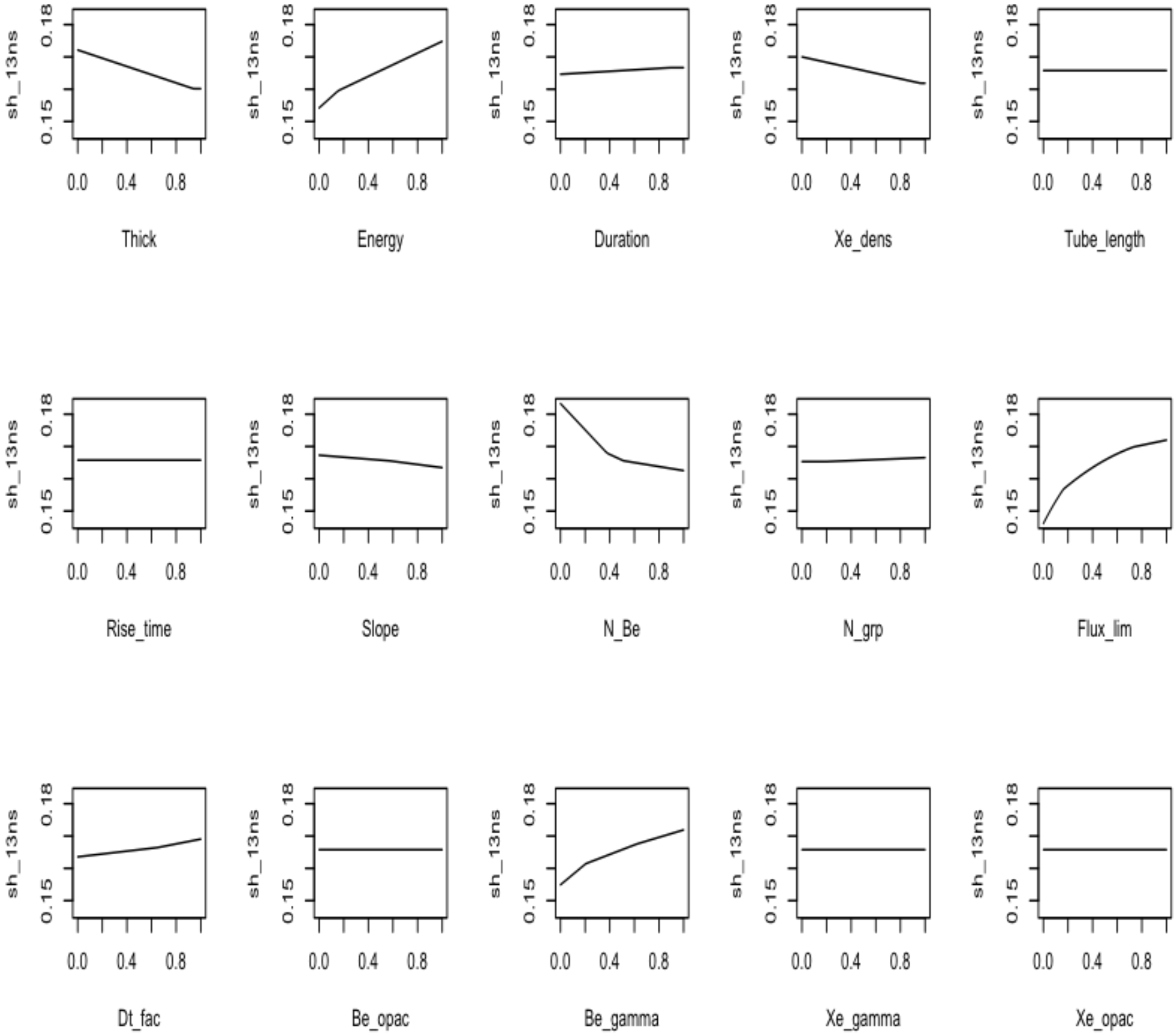


Figure 2.15: The 1-way marginal plots based on EMARS model on the Hyades data.

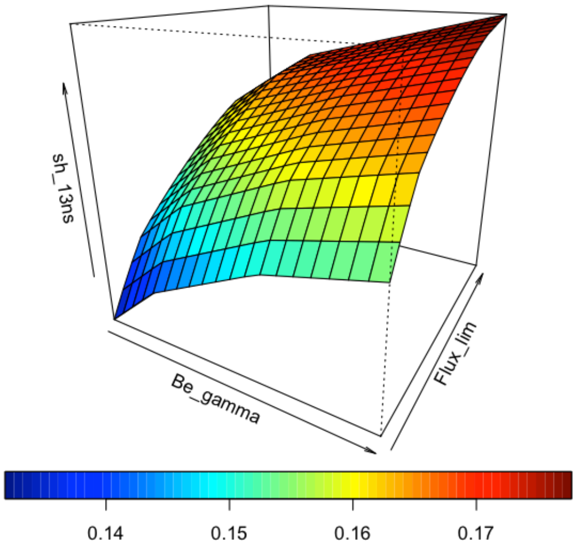


Figure 2.16: One example of the 2-way marginal plots based on EMARS model on the Hyades data. The surface plotted here is shock.location vs Be.gamma and Flux.lim parameter.

However, when we augmented the inert variables, the performance of EMARS stands out.

In the Piston-simulator study, from the marginal plots using EMARS (one-way marginals in Figure 2.5 and two-way marginal in Figure 2.6), we can see the underlying output-input relationship is very smooth. Also, it seems the first four inputs M, S, V_0, K dominate the variation of output, while effects of the others are relatively flat across their input domain. Figure 2.7 and 2.8 shows the residual plots of the 5 models and the results are consistent with the RMSE comparison in Table 2.3.

The OTL study gives similar story as the piston simulator study. Also notice that the performance of MART is not competitive, as the study involves very smooth functions. Figure 2.9-2.13 shows how the marginal plots and variable screening results

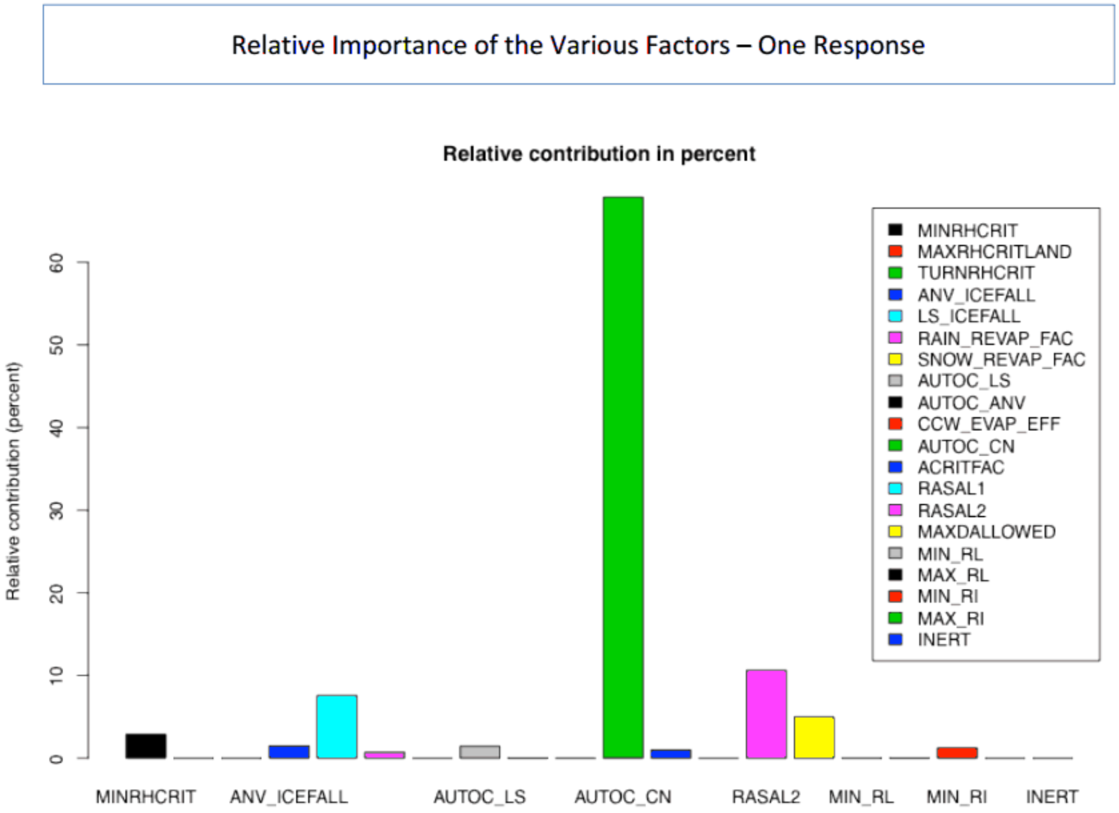
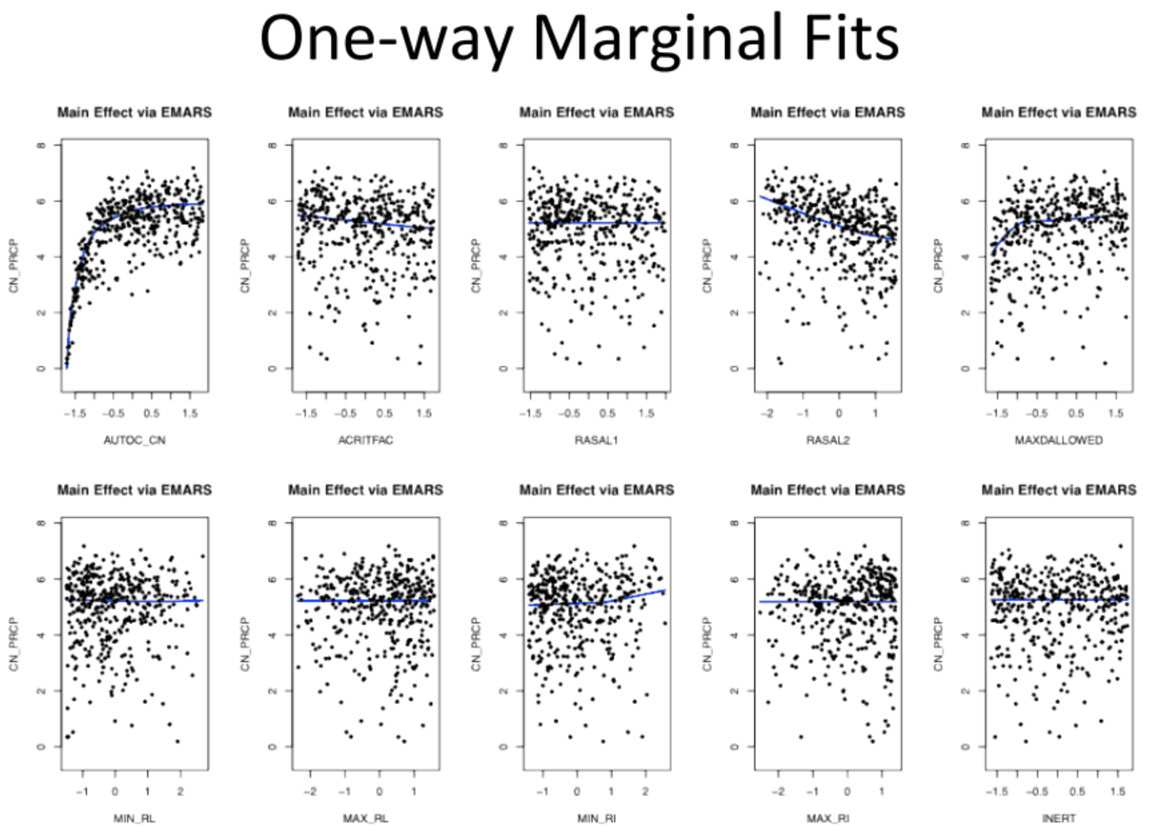


Figure 2.17: The relative variable contribution of the 20 inputs in the climate data study using EMARS model.



One-way Marginal Fits

Figure 2.18: The 1-way marginal sensitivity plots of the first 10 inputs in the climate data study using EMARS model.

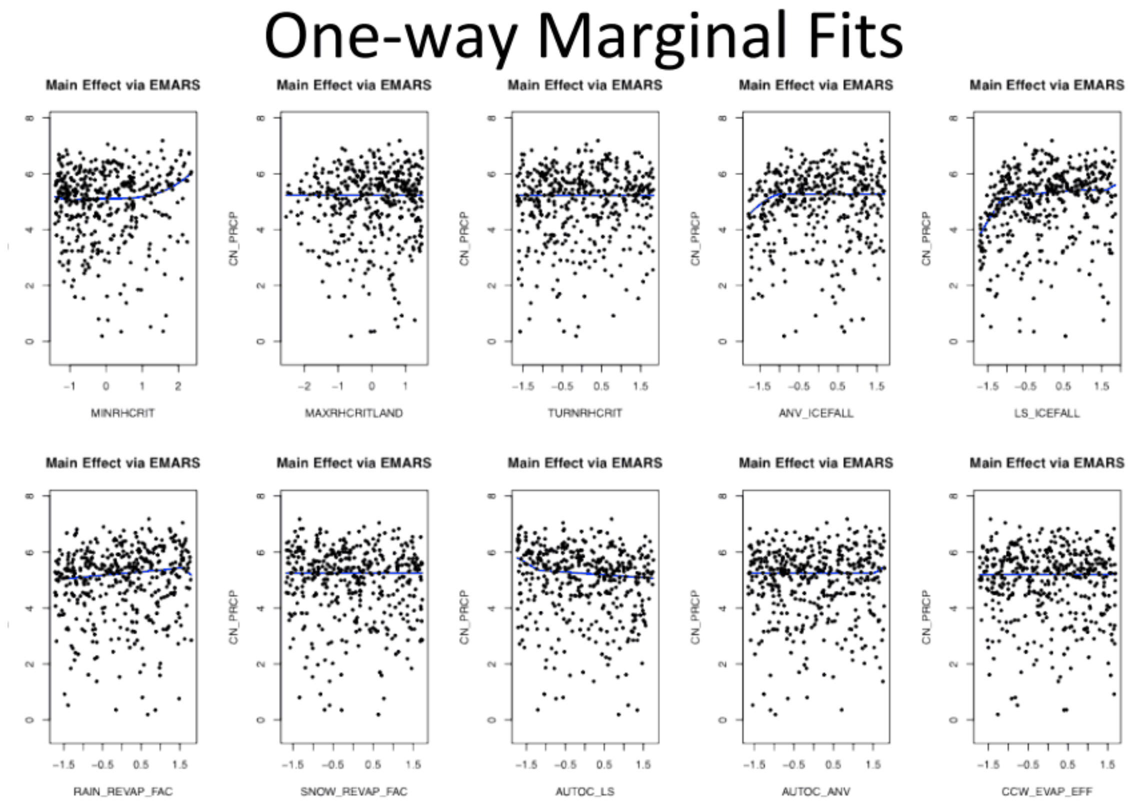


Figure 2.19: The 1-way marginal sensitivity plots of the second 10 inputs in the climate data study using EMARS model.

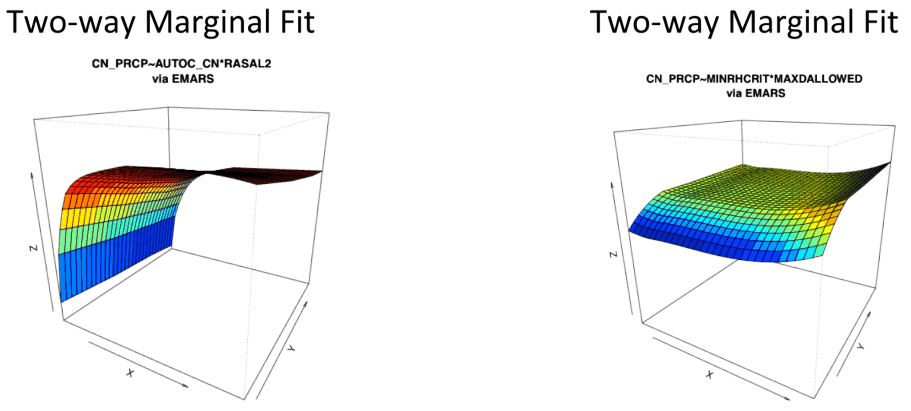


Figure 2.20: The 2-way marginal sensitivity plots of 2 pairs of most important inputs in the climate data study using EMARS model.

look like in this case.

On the 512 Hyades run data study and the climate data study, EMARS again does better than the others. SS-ANOVA fails to converge in this case because of the relatively large dimension: $d = 15$. Figure 2.15 shows the one-dimensional effects of each individual input, and we can see how the output interacts with those inputs. Figure 2.16 shows one example of the two-way marginal plots, $shock.location$ vs $Be.gamma$ and $Flux.lim$.

One interesting point we can see is that out of the 15 inputs, only 7 inputs have realistic impacts on the response, while the other 8 inputs stay relatively flat. This can again be confirmed in Figure 2.14, where we calculate the relative contribution of each individual inputs to the output. We used the permutation method to calculate the marginal contributions: we start with $RMSE$ e_0 on the training data, and to evaluate the contribution of each input, we randomly permute this input while keep the other inputs unchanged, so that the effect of this particular input would be depressed. Then we compute the $RMSE$ on the permuted data e_i, and use the difference as the contribution of that particular input $c_i = r_i - r_0$. What has been plotted in figure 6 the relative contribution, where we converted c_i into percentages: $\tilde{c}_i = \frac{c_i}{\sum_i c_i}$.

One additional comparison we have done is to see how well each method can screen out the purposely added "inert" variables. And the results in Table 2.7 shows that the EMARS is quite effective in screening inert variables out, while the benchmark GaSP model tend to be impacted by those noise inputs.

Table 2.7: Comparison of of inert variable screening abilities for 6 emulators. The table below shows summation of relative contributions of inerts.

Data	MARS	EMARS	SSANOVA	MART	B-GaSP	SS-Prod
Piston with inerts	0.0002	**0.0001**	NA	0.0003	0.0031	0.0003
OTL with inerts	0.0003	**0.000**	NA	0.0005	0.0042	0.0006

2.5 Summary

In this study, we compared the performance of six different statistical emulators: MARS, EMARS, MART, Bayesian GaSP, SS-Prod on a wide range of generated functions as well as illustrative applications. The six methods included the expanded MARS algorithm and the smoothing spline with product kernel. EMARS is a modified version of MARS to include polynomial terms such as $[x_i^2 - u]_+, [x_j^3 - v]_+ ...$ and their interactions. We see that EMARS does quite well in a variety of situations; it not only inherits the adaptivity of the MARS model, but is enriched by the additional basis components. The proposed SS-Prod model is a modification on the SS-ANOVA with a product exponential kernel function. It inherits the good parameter estimation using the the same generalized cross validation method, but avoids the difficulty that SSANOVA faces when the dimension of the problem goes up.

On the generated functions studies, we came up with 11 scenarios that covers a wide range of function characteristics: additivity, lower order interaction, higher order interaction, locally smooth and local interaction, the complexity due to inert variables and higher dimensional problem with sparse inputs. We also compared those 6 emulators on 4 known computer codes data. From the comparison results, we see that the EMARS does well in general. It outperforms GaSP model in most of the cases. The advantage of EMARS is especially high in cases where challenging local features exist, and in higher dimensional cases. EMARS appears to be able to do well in the presence of inert variables while the GaSP seems to be affected in such situations. SS-Prod has slightly better prediction result compared with GaSP. SSANOVA can perform quite well in lower dimensional problem, but does not scale up with dimension well. MART does not do well with smooth functions, but it can be very useful when the underlying problem involves heavy local features. Overall, in terms of computational efficiency and predictive performance, EMARS model stands out as a good alternative.

CHAPTER III

Calibration analysis of computer experiments

In this part of thesis, we focus on the calibration analysis of large-scale computational models. Calibrations problems have a long history and early studies had focused on simple regression models that arise in measurement standards. The calibration problem in our context involves complex models that are non-monotone and with many-to-one relationships. So the calibration problem, which involves inverse mappings, are challenging. Although the calibration problem has been studied in this context in the literature, there are several important questions that have not been addressed adequately. Perhaps the most important one is identifiability. We discuss the different kinds of identifiability issues that can arise and then developed several conditions that can be used to test for the existence of non-identifiability. Using numerical methods, we illustrated how to implement such conditions in several numerical examples as well as a case study.

3.1 Introduction

We had noted in the Introduction that large-scale computational models are used in applications where real physical experiments are impossible or difficult to conduct. For the purposes of calibration, however, we need some results from physical experiments against which the computational model and the calibration parameters can be calibrated.

So we denote y^f as the results from field experiments and y^c as the results from computer experiments. The inputs of computer experiments and the field experiments, however, are not exactly the same. Beside common inputs shared by both field experiments and computer experiments, which are usually refereed as variable inputs or physical inputs, there are some additional inputs in computer experiments, usually called calibration inputs. Specifically, a computer model output $y^c(x, \theta) = f(x, \theta)$ depends on both physical inputs x and calibration inputs θ, but a field output $y^f(x) = f(x, \theta^*) + \epsilon$ only depends on physical inputs x. Those θ^*s are fixed, unknown, uncontrollable parameters in field experiments, but θs can vary as inputs in computer models. The goal is to combine both computer data and field data to calibrate the computer model by conducting inference on calibration parameters θ^*, make prediction of future field experiments and quantify the uncertainty of such a prediction. This is known as computer model calibration, which has been discussed extensively in recent years [37] [34] [35] [3] [30].

The popular approach to deal with computer model calibration is using Gaussian Spatial process models (GaSP). Kennedy and O'Hagan [37] proposed a Bayesian method based on GaSP for calibration. The basic idea is to treat the computer data and field data as the realizations of two correlated Gaussian Processes. The Bayesian approach combines both sources of data to do both prediction and calibration by accounting for multiple source of uncertainties. Higdon et al. [35] have extended the Bayesian calibration method into cases with multiple outputs. Bayarri et al. [3] extended the method to problems with functional outputs. As an alternative to the Bayesian method, we will show later that one can also tackle the calibration problem in a frequentist's framework.

Despite extensive discussion of statistical approaches in computer model calibration, the question of whether or not an underlying calibration problem is identifiable is still difficult to ascertain. As the functional relationship between the output of a computer model and calibration parameters is generally unknown, there could be cases when multiple sets (may be infinite) of calibration parameters that lead to the same field output. This phenomenon is described as non-identifiability and we will formally defined it in later sections. For instance, consider a simple example where the computer model output is $f(x_1, x_2, \theta_1, \theta_2) = x_1 + x_2 + \theta_1 - \theta_2$ and the experimental process $y^f(z) = f(z_1, z_2, 0.2, 0.8) + \epsilon$, where both x, z and θ are in $[0, 1]$, and ϵ is a random error. It is not difficult to see that in this case, θ_1 and θ_2 can not be calibrated as there are infinitely many θ_1 and θ_2 pairs such that $\theta_1 - \theta_2$ is equal to 0.6. Figure 3.1 shows the results using two different calibration methods, which confirm the conclusion.

In order to tackle this problem, we first formally define parameter identifiability in computer models and then developed several conditions to test the existence of such issues. Using numerical methods, we illustrated how to implement the conditions in several simulation examples as well as a case study. The outline of this paper is as follows. In Section 3.2, we do a brief review of Bayesian Gaussian Process calibration method for computer experiments. In Section 3.3, we discuss an alternative way to tackle the calibration problem in a freqentist approach. In Section 3.4, we formally discuss the potential identifiability issue and developed statistical procedures for checking parameter non-identifiability. In Section 3.5, we compare the EMARS calibration method versus the widely used Bayesian GaSP calibration approach in identifiable cases. In the last section, we conducted a real application case study on the CRASH calibration problem.

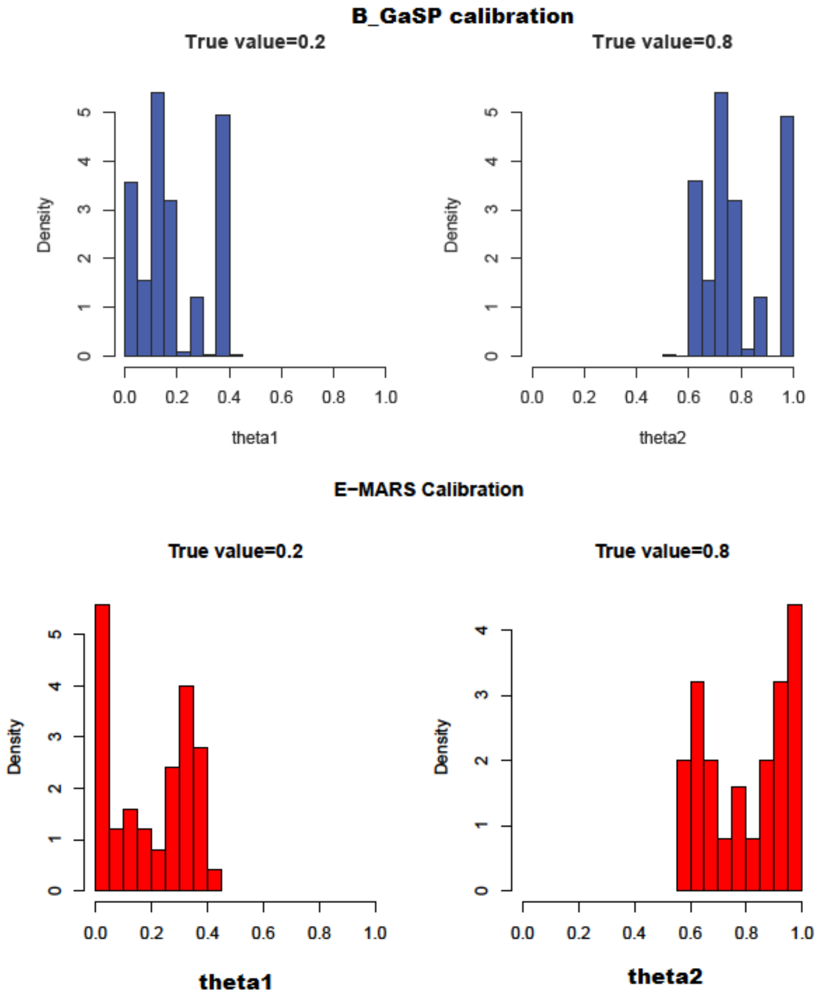


Figure 3.1: The upper plot shows the posterior distribution of the calibration parameter using the Bayesian GaSP method, in a simulation study with $f(x_1, x_2, \theta_1, \theta_2) = x_1 + x_2 + \theta_1 - \theta_2$ and $Y^f(z_1, z_2) = f(z_1, z_2, 0.2, 0.8) + \epsilon$, where $\epsilon \sim N(0, 0.1^2)$ and all inputs are in $[0, 1]$. The lower panel shows the calibration result using EMARS based bootstrap samples. As we can see, θ_1 and θ_2 can not be calibrated in this case.

3.2 A review of the Bayesian GaSP calibration method

As described in the introduction, in a lot of scientific studies where both computer models and field experiments are available, there is a need to use both data to calibrate the computer model by estimating a set of calibration parameters. Formally, we partition the inputs of a computer model into $\{x, \theta\}$ where $\{\mathbf{x}\} = \{\mathbf{x_1}, ..., \mathbf{x_p}\}$, the p-dimensional regular (physical) input parameters, and $\{\theta\} = \{\theta_\mathbf{1}, ..., \theta_\mathbf{k}\}$, the k-dimensional calibration parameters. In field data, we use $\{z_1, ..., z_p\}$ to denote the physical inputs and $\{\theta_1^*, \ldots, \theta_k^*\}$ to denote the true unknown calibration parameters in the physical process. The outputs of simulator runs and field results are denoted as y^c and y^f respectively.

Kennedy and O'Hagan [37] (2001) proposed a Bayesian GaSP calibration method, where y^c and y^f are considered as realizations of two correlated Gaussian processes Y^c and Y^f:

$$Y^c(x, \theta) = \eta(x, \theta) \tag{3.1}$$

$$Y^f(z) = \eta(z, \theta^*) + \delta(z) + \epsilon \tag{3.2}$$

where η is a stationary Gaussian process with constant mean μ and covariance generated by a radial basis kernel, $\delta(z)$ is used to characterize the possible discrepancy between computer simulator and field runs and is also assumed be a Gaussian process with mean 0 and a covariance structure similar to η, ϵ is assumed to be i.i.d. observational random error with $N(0, \sigma^2)$, and η, δ, ϵ are assumed to be independent of each other.

In general, the covariances are given as:

$$\mathrm{Cov}(Y^c(x,\theta), Y^c(x',\theta')) = \frac{1}{\lambda_c}\prod_{j=1}^{p} e^{-\tau_{cj}(x_j - x'_j)^2} \prod_{j'=1}^{k} e^{-\tau_{\theta j'}(\theta_{j'} - \theta'_{j'})^2} \tag{3.3}$$

$$\mathrm{Cov}(Y^c(x,\theta), Y^f(z)) = \frac{1}{\lambda_c}\prod_{j=1}^{p} e^{-\tau_{cj}(x_j - z_j)^2} \prod_{j'=1}^{k} e^{-\tau_{\theta j'}(\theta_{j'} - \theta^*_{j'})^2} \tag{3.4}$$

$$\mathrm{Cov}(Y^f(z), Y^f(z')) = \frac{1}{\lambda_c}\prod_{j=1}^{p} e^{-\tau_{cj}(z_j - z'_j)^2} + \frac{1}{\lambda_\delta}\prod_{j=1}^{p} e^{-\tau_{\delta j}(z_j - z'_j)^2} + \sigma^2 I(z = z') \tag{3.5}$$

Writing all the hyper parameters $\{\lambda_c, \lambda_\delta, \sigma^{-2}, \tau_{cj}, \tau_{\theta j'}, \tau_{\delta j}, j = 1..p, j' = 1....k\}$ as ψ, and combing $\{y^c_i, y^f_{i'}, i = 1...n, i' = 1...m\}$ as one vector d, the log-likelihood is given as

$$l(d|\psi, \mu, \theta^*) \propto -\frac{1}{2}\log|\Sigma| - \frac{1}{2}(d-\mu)'\Sigma^{-1}(d-\mu) \tag{3.6}$$

where

$$\Sigma = \begin{bmatrix} \Sigma_{cc}(\psi) & \Sigma_{cf}(\psi,\theta) \\ \Sigma'_{cf}(\psi,\theta) & \Sigma_{ff}(\psi) \end{bmatrix}$$

and Σ_{cc}, Σ_{cf} and Σ_{ff} are calculated by (3.3), (3.4) and (3.5) respectively. By imposing appropriate priors π on the parameters $\{\psi, \mu, \theta^*\}$, the posterior log-likelihood is given as

$$l(\{\psi, \mu, \theta^*\}|d) \propto -\frac{1}{2}\log|\Sigma| - \frac{1}{2}(d-\mu)'\Sigma^{-1}(d-\mu) + \log(\pi(\psi, \mu, \theta)) \tag{3.7}$$

For priors, normal or uniform priors are often assumed for θ^* and μ. For hyper parameters in ψ, typical prior choices are $\lambda_c \sim Gamma(10, 10)$, $\sigma^{-2} \sim Gamma(10, 0.1)$ (Note that in practice, inputs are usually scaled to be in $[0, 1]$); for those τ's, beta priors are often assumed in the following fashion:

$$\pi(\tau) \propto \prod_j (1 - e^{-\tau_j})^{-0.5} e^{-\tau_j}, \tau_j \geq 0$$

A fully Bayesian analysis would use MCMC samples to construct posterior distribution from (3.7) for the calibration parameter θ^* and further construct the predictive distribution for a future field run of $y^f(z^*)$.

3.3 A frequentist's approach for computer model calibration

As an alternative to the Bayesian Gaussian process calibration method, one may consider to adopt a frequentist's approach. First, suppose the underlying function of the computer model is: $Y^c(x, \theta) = f(x, \theta)$ while the field data come from the process

$$Y^f(z) = f(z, \theta^*) + \epsilon_f$$

where the errors ϵ_f is assumed to be i.i.d. random errors.

In an ideal case, where we assume the computer model is efficient enough to produce any output at a given input almost instantly (cheap code), then the calibration problem boils down to find an optimal estimate $\widehat{\theta^*}$:

$$\widehat{\theta^*} = \underset{\theta^* \in \Theta}{\operatorname{argmin}} \sum_{i=1}^{m} |y_i^f - f(z_i, \theta^*)|^2 \tag{3.8}$$

However, in practice, we typically have to rely on statistical emulators to first estimate the unknown function of the computer model $f(x, \theta)$. For instance, we can use the popular Gaussian kriging method, or the EMARS approach that we discussed in Chapter 2 to estimate $f(x, \theta)$. For details of those emulators, please refer to chapter 2 of this thesis. Now given the fitted function $\hat{f}(x, \theta)$, and the physical data y^f, we may optimize the following criteriion to find the optimal calibration parameter $\widehat{\theta^*}$:

$$\widehat{\theta^*} = \underset{\theta^* \in \Theta}{\operatorname{argmin}} \sum_{i=1}^{m} |y_i^f - \hat{f}(z_i, \theta^*)|^2 \tag{3.9}$$

After obtaining $\widehat{\theta^*}$, the prediction of the physical process at an input x^* is given by:

$$\hat{y}(x^*) = \hat{f}(x^*, \widehat{\theta^*}) \tag{3.10}$$

To estimate the variance of $\widehat{\theta^*}$ and $\hat{y}(x)$, we may use bootstrap, i.e. randomly select B bootstrap samples from the computer data as well as the physical data; for each bootstrap sample, we obtain an estimate of the calibration parameter and the corresponding prediction for $y(x^*)$, thus variances and confidence intervals can be created using bootstrap estimates.

3.4 Calibration parameter identifiability

In the above sections, we reviewed two types of approaches for computer model calibration: the Bayesian GaSP calibration method and the frequentist's approach. However, one important prerequisite for these methods to work is that the underlying calibration problem is identifiable, i.e. the calibration parameter of interest can be calibrated. In this section, we focus on this identifiability issue, and we start with some definitions.

3.4.1 Definition of identifiability in calibration

Definition 1: Identifiability in computer model calibration. Consider a computer model with $y^c = f(x, \theta)$, where $x \in X$ is the physical input and $\theta \in \Theta$ is the calibration input. If $f(x, \theta) = f(x, \theta')$ for all $x \in X$ implies $\theta = \theta'$, we call the computer model calibration problem identifiable; otherwise the underlying calibration problem is non-identifiable.

Under this definition, we see that for non-identifiable calibration problems, there exist at least two distinct sets of calibration parameters $\theta \in \Theta$ and $\theta' \in \Theta$, such that $f(x, \theta) = f(x, \theta')$ for all $x \in X$. For instance, the example we discussed in the introduction, i.e. $f(x_1, x_2, \theta_1, \theta_2) = x_1 + x_2 + \theta_1 - \theta_2$, would be such a non-identifiable problem. In fact, for this particular example, there exist infinitely many calibration parameters θ such at they all lead to the same computer model $f(x, \theta)$ (as a function of x). We define this phenomenon as intrinsic non-identifiability.

Definition 2: Intrinsic non-identifiability in computer model calibration. For a calibration problem with $k > 1$ calibration parameters $(\theta_1, ...\theta_k)$, if there exists a transformed set of parameters $\beta = (\beta_1(\theta), \ldots, \beta_q(\theta))$ with $q < k$ such that $f(x, \theta) = f(x, \beta)$ for all x, then we call the calibration problem intrinsically non-identifiable.

Though the intrinsically non-identifiable problems cover a large portion of all the non-identifiable calibration problems, there are other types of non-identifiability such as $f(x, \theta) = x + \theta^2$, where $\theta \in R$. Note that for this type, there exist multiple but a finite number of calibration points which lead to the same computer model $f(x, \theta)$. We call this type as non-identifiability caused by “multiple solutions”. Further, there are problems where the two types of non-identifiability mix together. In the coming sections, we develop conditions to check the existence of intrinsic non-identifiability and also give some empirical guidelines for checking non-identifiability caused by multiple solutions.

3.4.2 A sufficient and necessary condition on intrinsic non-identifiability

Based on the definition of intrinsic identifiability, it is not difficult to see that identifiability is related to partial derivatives of $f(x, \theta)$ with respect to θ. Catchpole and Morgan [9] proved a theoretical result on checking parameter redundancy for a family of complicated models that originated in biological research. Motivated by their result, we derive the following sufficient and necessary condition on intrinsic non-identifiability for computer model calibration.

Proposition 1: A sufficient and necessary condition on intrinsic non-identifiability.

Consider a computer model with $y^c = f(x, \theta)$, where $x \in X$ is the physical input and $\theta \in \Theta$ is the calibration input. Suppose the partial derivatives of f with respect to θ exist on $X \times \Theta$. Let

$$D(x, \theta) = \left\{ \frac{\partial f(x, \theta)}{\partial \theta_1}, \frac{\partial f(x, \theta)}{\partial \theta_2}, \ldots, \frac{\partial f(x, \theta)}{\partial \theta_k} \right\}.$$

Then the intrinsic non-identifiability of the calibration problem is equivalent to: there exists a k-dimensional non-zero vector $\lambda(\theta)$, which only depends on θ, such that

$$\lambda'(\theta) D(x, \theta) \equiv 0 \text{ for all } x \in X \text{ and } \theta \in \Theta.$$

The proof of Proposition 1 resembles that given by Catchpole and Morgan [9]: the necessity comes directly from the differential chain rule, and the sufficiency comes from the general solution of the first order Lagrange linear partial differential equations. We omit the details.

To better understand this condition, we consider two examples.

- Example 1: $y^c = f(x,\theta) = f_x(x) + f_\theta(\theta)$, where the effects of x and θ are completely additive, and the number of calibration parameters $k > 1$. Since $D = (f'_{\theta_1}, ...f'_{\theta_k})$ does not involve x, it is not difficult to see the existence of such $\lambda(\theta)$. For example, when $k = 2$, one may let $\lambda_1(\theta) = 1$ and $\lambda_2(\theta) = -f'_{\theta_1}/f'_{\theta_2}$; when $k > 2$, one may keep the same λ_1 and λ_2 while set $\lambda_3 = \ldots = \lambda_k = 0$.

- Example 2: $f(x,\theta) = \sum_j \theta_j \phi_j(x)$. In this case, $D(x,\theta) = (\phi_1(x), \phi_2(x)....\phi_k(x))$, and the existence of $\lambda(\theta)$ demands that these basis functions be linearly dependent, which is what one would expect.

3.4.3 A numerical approach to implement the necessary and sufficient condition

Recall that the sufficient and necessary condition of intrinsic parameter non-identifiability boils down to the existence of non-zero $\lambda(\theta) = (\lambda_1(\theta), \ldots, \lambda_k(\theta))$ such that

$$\sum_{j=1}^{k} \lambda_j(\theta) D_j(x,\theta) = 0 \tag{3.11}$$

for all $x \in X$ and $\theta \in \Theta$, where $D_j(x,\theta) = \frac{\partial f(x,\theta)}{\partial \theta_j}$, $j = 1, \ldots, k$.

Since the functional form of $f(x,\theta)$ is unknown, checking (3.11) directly is challenging. In this section, we propose a numerical approach to check whether (3.11) holds. The idea is to find a $\lambda(\theta)$ such that the sum of $\left(\sum_{j=1}^{k} \lambda_j(\theta) D_j(x,\theta)\right)^2$ over a grid of points is minimized. If the resulting minimum is close to zero, it implies the existence of a $\lambda(\theta)$ such that (3.11) holds, otherwise, such a $\lambda(\theta)$ probably does not exist.

Specifically, given n design points $(x_1, \theta_1), \ldots, (x_n, \theta_n)$ we consider to minimize

$$\min_{\lambda_j(\theta_i)} \sum_{i=1}^{n} \left(\sum_{j=1}^{k} \lambda_j(\theta_i) D_j(x_i, \theta_i) \right)^2 \tag{3.12}$$

subject to

$$\sum_{j=1}^{k}\sum_{i=1}^{n}\lambda_j(\theta_i)^2 = 1 \tag{3.13}$$

Note that (3.13) ensures the vector $\lambda(\theta)$ will not be a 0 vector.

In order to allow for flexible $\lambda(\theta)$, we consider to use the radial basis kernel functions. Specifically, we model $\lambda(\theta)$ with $\lambda_j(\theta) = \sum_{i=1}^{n}\alpha_{ji}K(\theta, \theta_i)$ (for notational simplicity we did not include the intercept term), where $K(u, v) = \exp(-\sum_{j=1}^{k}\tau_j(u_j - v_j)^2)$ and α_{ji}'s are unknown. Then (3.12) and (3.13) can be transformed to

$$\min_{\alpha_1,\ldots,\alpha_p}\sum_{i=1}^{n}\left(\sum_{j=1}^{k}K_i'\alpha_j D_{ij}\right)^2 \tag{3.14}$$

subject to

$$H = \sum_{j=1}^{k}\alpha_j' K\alpha_j - 1 = 0 \tag{3.15}$$

where K_i denotes the ith column of the kernel matrix $K = [K(\theta_i, \theta_{i'})]$. The gradient of the objective function can be computed as

$$G_j = 2\sum_{i=1}^{n}\left(\sum_{j=1}^{k}K_i'\alpha_j D_{ij}\right)D_{ij}K_i',\ j = 1, \ldots, k \tag{3.16}$$

Similarly, the Jocobian is:

$$J_j = \frac{\partial H}{\partial \alpha_j} = 2K\alpha_j\ j = 1, ...k \tag{3.17}$$

With the Jacobian and the gradient, one can solve for α_j, $j = 1, \ldots, k$ using many existing optimization packages, including the the package “alabama” in R.

Evaluate $\hat{\lambda}(\theta)$

Let $\hat{\alpha}$ denote the minimizer of (3.14), and $\hat{\lambda}_{ij} = \sum_{i'=1}^{n} \hat{\alpha}_{ji'} K(\theta_i, \theta_{i'})$. To determine whether the conditions in (3.11) are met, we define a discrete (and normalized) version of $\lambda'(\theta)D$ as follows:

$$\hat{C} = \frac{1}{n} \sum_{i=1}^{n} \frac{(\sum_{j=1}^{k} \widehat{\lambda_{ij}} D_{ij})^2}{\sum_{j=1}^{k} [\widehat{\lambda_{ij}}]^2 \cdot \sum_{j=1}^{k} [D_{ij}]^2} = \frac{1}{n} \sum_{i=1}^{n} [\cos(\gamma_i)]^2 \tag{3.18}$$

where γ_i can be considered as the angle between the vectors $\hat{\lambda}_i$ and D_i. Note that $\hat{C}$ ($0 \leq \hat{C} \leq 1$) is a normalized quantity so that it is invariant to the scale of inputs and outputs. Intuitively if the calibration problem is non-identifiable, one would expect $\hat{C}$ to be small (close to 0); on the other hand, if $\hat{C}$ is sufficiently large, the calibration problem is probably identifiable.

To "test" whether $\hat{C}$ is large enough, we build a reference distribution for $\hat{C}$. Specifically we introduce two additional calibration inputs u_1 and u_2, and modify the output as follows:

$$f^*(x, \theta, u_1, u_2) = f(x, \theta) + b(u_1 + u_2) \tag{3.19}$$

where b is a scalar. It is clear that the parameters set $(\theta_1, \ldots, \theta_k, u_1, u_2)$ are non-identifiable in the calibration setting. Nevertheless, we may apply (3.14) under this new setting, with $D^* = [D, b, b]$ and record the resulting C^* (which should be close to 0). We repeat the procedure for different values of b (e.g. randomly drawn from the uniform distribution $U[-1, 1]$), and obtain different C^* values, e.g. $\{C_1^*, \ldots, C_M^*\}$, where B is the number of random draws for b. Let

$$\hat{p} = \frac{\sum_{\ell=1}^{B} I(C_\ell^* \leq \hat{C})}{B} \tag{3.20}$$

Note that if the calibration problem is non-identifiable, the value of $\hat{p}$ would be close to 0, while if the value of $\hat{p}$ is close to 1, one may conclude the calibration problem as identifiable. Further, note that $b(u_1 + u_2)$ is not the only functional form that can be appended to $f(x, \theta)$. In fact, any general smooth function $g(u_1, u_2)$ that does not involve x could serve the purpose. For instance, we may also consider the following modification to $f(x, \theta)$:

$$f^*(x, \theta, u_1, u_2) = f(x, \theta) + b(u_1 u_2) \tag{3.21}$$

where b can be drawn from $U[-1, 1]$. In this case, the corresponding $D^* = [D, bu_2, bu_1]$.

3.4.4 Simulation studies for the sufficient and necessary condition

In this subsection, we use simulation examples to illustrate the implementation of the sufficient and necessary condition in Proposition 1 through the numerical approach described above. Specifically, we consider 6 functions:

- A: $y = 1 + x_1^2 + x_2 + x_3(\theta_1^2 + \theta_2)$
- A': $y = 1 + x_1^2 + x_2 + x_3(\theta_1^2 + \theta_2) + \theta_2$
- B: $y = 1 + x_1^2 + x_2 + \exp[-x_3(\theta_1^2 + \theta_2)]$
- B': $y = 1 + x_1^2 + x_2\theta_2 + \exp[-x_3(\theta_1^2 + \theta_2)]$
- C: $y = 1 + x_1^2 + \theta_3(x_2 + x_3) + \exp[-x_4(\theta_1^2 + \theta_2)]$
- C':$y = 1 + x_1^2 + \theta_3(x_2 + x_3) + \exp[-x_4(\theta_1^2 + \theta_2)] + \theta_1$

In all 6 functions, regardless of the dimension, we generate 100 Latin hypercube design points on either $[0, 1]^5$ or $[0, 1]^6$ for each of the 6 functions. Clearly we see that A, B, C are non-identifiable, while A', B', C' are identifiable. There are two calibration parameters θ_1, θ_2 in cases $A-B'$ and three calibration parameters in cases $C-C'$.

First, we assume the ideal situation where we can evaluate any inputs with no cost. Thus we can compute the matrix D using numerical derivative with a sufficiently small Δ, i.e.

$$\frac{\partial f}{\partial \theta_j} := \frac{f(x, \theta_{-\{j\}}, \theta_j + \Delta) - f(x, \theta_{-\{j\}}, \theta_j)}{\Delta}, \ j = 1, \ldots, k \tag{3.22}$$

Given D, we solve the optimization problem (3.14) and calculate the criterion $\hat{C}$ in (3.18). To determine if $\hat{C}$ is sufficiently close to 0, we introduce two reference variables u and v, and modify the function $f(x, \theta)$ as $f(x, \theta) + b(u + v)$ or $f(x, \theta) + b(uv)$. We generate $B = 100$ values of b from the uniform distribution $U[-1, 1]$, calcuate C^*'s and finally use equation (3.20) to compute $\hat{p}$. Table 3.1 shows the values $\hat{C}$ and $\hat{p}$ for each of the 6 functions based on additive reference and product reference.

In practice, we usually do not have the “cheap code” and have to rely on statistical emulators such as GaSP or EMARS to calculate D. Table 3.2 shows the results using EMARS and table 3.3 shows the results using GaSP. From these 3 tables (Tables 3.1-3.3), we see that the estimated $\hat{p}$ is quite consistent with the truth. While EMARS and GaSP give similar results, the result based on the numerically calculated derivative matrix D (ideal situation) is the best. This is not surprising, as estimation based on emulators produced another layer of approximation. Comparing additive reference with the product reference, we can see that C^* based on the product reference is in general larger, so that the corresponding $\hat{p}$ is smaller. This implies that if one is conservative on claiming identifiability, the product reference is preferred.

Further, we also investigated how random errors on y^c could impact the results. This is relevant because even many computational models can be viewed as deterministic, there are cases where the output suffers from errors. For cases C and C', we introduce an error term $\epsilon \sim N(0, \sigma^2 = \gamma \cdot var(y^c))$, where γ varies from 0.01 to 0.1. Figure 3.2 shows the estimated $\hat{p}$ vs σ. The result is reasonable, i.e., as the error term gets larger, the problem becomes more difficult and the estimated $\hat{p}$ becomes less consistent with the truth.

Table 3.1: Results based on numerical derivatives. The upper table uses the additive reference and the lower table uses the product reference.

case	$\hat{C}$	$Avg(C^*)$	$\hat{p}$	Identifiable(truth)?
A	2.2E-10	3.3E-10	7%	No
A'	4.57E-2	5.3E-10	100%	Yes
B	4.7E-8	5.9E-8	13%	No
B'	1.34E-2	7.3E-8	100%	Yes
C	3.2E-7	3.3E-7	19%	No
C'	1.2E-3	5.5E-7	98%	Yes

case	$\hat{C}$	$Avg(C^*)$	$\hat{p}$	Identifiable(truth)?
A	2.2E-10	4.3E-10	6%	No
A'	4.57E-2	8.3E-10	100%	Yes
B	4.7E-8	9.1E-8	11%	No
B'	1.34E-2	1.2E-7	100%	Yes
C	3.2E-7	6.9E-7	15%	No
C'	1.2E-3	8.01E-7	97%	Yes

Table 3.2: Results based on EMARS derivatives. The upper table uses the additive reference and the lower table uses the product reference.

case	$\hat{C}$	$Avg(C^*)$	$\hat{p}$	Identifiable(truth)?
A	3.70E-08	4.1E-8	11%	No
A'	8.87E-02	5.5E-7	98%	Yes
B	8.61E-06	9.93E-6	21%	No
B'	2.71E-03	1.32E-6	93%	Yes
C	1.09E-06	1.88E-6	29%	No
C'	3.50E-03	8.01E-6	88%	Yes

case	$\hat{C}$	$Avg(C^*)$	$\hat{p}$	Identifiable(truth)?
A	3.70E-08	5.2E-8	9%	No
A'	8.87E-02	6.3E-7	97%	Yes
B	8.61E-06	9.99E-6	19%	No
B'	2.71E-03	3.32E-6	91%	Yes
C	1.09E-06	1.95E-6	26%	No
C'	3.50E-03	9.31E-6	87%	Yes

Table 3.3: Results based on GaSP derivatives. The upper table uses the additive reference and the lower table uses the product reference.

case	$\hat{C}$	$Avg(C^*)$	$\hat{p}$	Identifiable(truth)?
A	6.30E-07	8.2E-7	13%	No
A'	1.32E-01	4.87E-6	96%	Yes
B	4.52E-05	6.43E-5	23%	No
B'	4.31E-2	4.35E-05	94%	Yes
C	3.33E-05	5.82E-5	31%	No
C'	2.70E-02	6.19E-5	86%	Yes

case	$\hat{C}$	$Avg(C^*)$	$\hat{p}$	Identifiable(truth)?
A	6.30E-07	9.5E-7	12%	No
A'	1.32E-01	6.8E-6	95%	Yes
B	4.52E-05	8.8E-5	21%	No
B'	4.31E-2	7.30E-05	90%	Yes
C	3.33E-05	6.4E-5	28%	No
C'	2.70E-02	9.5E-5	83%	Yes

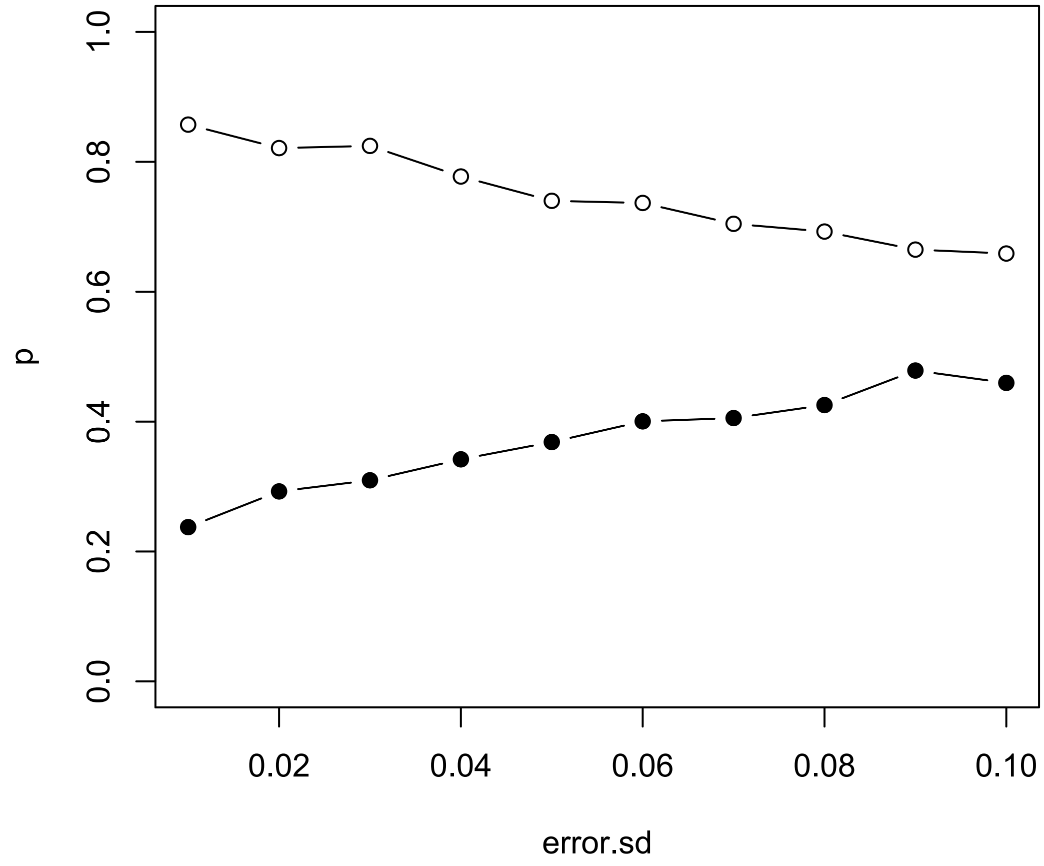


Figure 3.2: Impact of a random error term: estimated $\hat{p}$ vs the standard deviation of the error term. Solid points correspond to case C and circle points correspond to case C'.

3.5 A special case – a sufficient condition using additivity

In this section, we discuss a sufficient condition for checking calibration parameter non-identifiability. As one can see in example 1 of section 3.4.2, when there are more than one calibration parameters, additivity in physical input x and calibration input θ implies non-identifiability in computer model calibration. In fact, it is not required that x and θ are completely additive, as long as more than one calibration parameters are additive to x, the calibration problem is non-identifiable. We give a formal definition as follows.

Definition 3: Additivity of one calibration input to all physical inputs. Consider a computer model with $y^c = f(x, \theta)$, where $x \in X$ is the physical input and $\theta \in \Theta$ is the calibration input. Suppose the Hessian matrix $[\frac{\partial^2 f(x,\theta)}{\partial x_j \partial \theta_{j'}}]$ exists on $X \times \Theta$. For some $\theta_{j'}$, if

$$\frac{\partial^2 f(x, \theta)}{\partial x_j \partial \theta_{j'}} = 0$$

for all $j = 1, \ldots, p$, we call $\theta_{j'}$ additive to the physical inputs x.

Using this definition, we have the following sufficient condition on intrinsic non-identifiability.

Proposition 2: A sufficient condition using additivity. In a computer model calibration problem, if there are more than one calibration inputs that are additive to x, then the calibration problem is intrinsically non-identifiable.

To see why Proposition 2 holds, suppose θ_1 and θ_2 are additive to all physical inputs x, then we can choose $\lambda_1(\theta) = 1$ and $\lambda_2(\theta) = -f'_{\theta_1}/f'_{\theta_2}$, and set the rest of $\lambda(\theta)$ to be 0 (if the number of calibration parameters $k > 2$). Then the sufficient condition in Proposition 1 would hold, i.e.

$$\lambda'(\theta)D(x,\theta) \equiv 0 \text{ for all } x \in X \text{ and } \theta \in \Theta.$$

It should be noted that the condition in Proposition 2 is a special case of Proposition 1, however, the examination of individual additivity between $\theta_{j'}$ and x_j is useful in terms of understanding which subset of calibration parameters can not be identified due to their additivity to x.

3.5.1 A numerical approach to implement the special condition with additivity

To check the special condition in Proposition 2, we also rely on numerical approaches. Intuitively, we need to check whether the second order partial derivatives $\frac{\partial^2 f(x,\theta)}{\partial x_j \partial \theta_{j'}}$ are sufficiently small.

Given a grid of points (x_1, θ_1), ..., (x_n, θ_n), we define the following quantity to measure the significance of the interaction between x_j and $\theta_{j'}$:

$$I_{jj'} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{\partial^2 f(x,\theta)}{\partial x_j \partial \theta_{j'}} (x_i, \theta_i) \right|$$

If $I_{jj'}$ is sufficiently close to zero, it implies that x_j and $\theta_{j'}$ are nearly additive. To compute the second order partial derivatives, one may, again, use either the numerical derivatives (if the computer model is "cheap"), or a statistical emulator, such as the GaSP or EMARS.

To decide whether an $I_{jj'}$ is sufficiently small, similar to what has been done in section 3.4.3, we introduce an inert variable u (e.g. randomly generated from $U[0,1]$), and modify the computer output as follows:

$$f^*(x, \theta, u) = f(x, \theta) + u$$

For the modified computer data $\{f^*; x, \theta, u\}$, we may fit a statistical emulator, such as the GaSP or EMARS, and calculate the quantities $\{I_{x_j,u}, I_{\theta_{j'},u}\}$. Since u is additive to other inputs, we would expect $I_{x_j,u}$, $I_{\theta_{j'},u}$ to be close to zero. Thus we define the reference

$$I_u = \max\{I_{x_j,u}, I_{\theta_{j'},u}, j = 1, \ldots, p, j' = 1, \ldots, k\}$$

We repeat the procedure B times (with a different value of u for each time), obtain $I_u^1, \ldots, I_u^B$ and define

$$\hat{p}_{jj'} = \sum_{\ell=1}^{B} I(I_{jj'} - I_u^\ell < 0)/B,\ j = 1, \ldots, p, j' = 1, \ldots, k$$

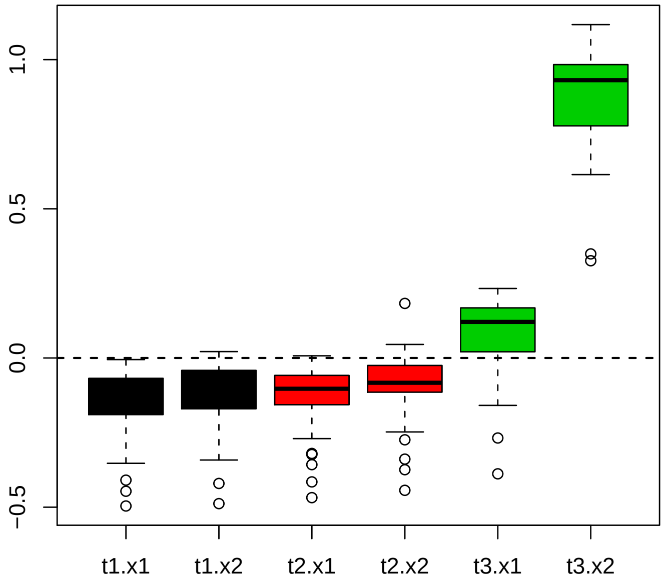
If the value of $\hat{p}_{jj'}$ is close to 1, it implies x_j and $\theta_{j'}$ are additive to each other; on the other hand, if $\hat{p}_{jj'}$ is close to 0, it implies x_j and $\theta_{j'}$ are non-additive.

3.5.2 Simulation studies on the special condition with additivity

In this section, we use simulation studies to illustrate the implementation of the additivity condition. We consider two examples:

- Case A: $f(x, \theta) = 0.2 + x_1 x_2^2 + \cos(x_1 + x_2) + 0.5 * \theta_3 \exp(-x_1) + 1.5 * \sin(\theta_3) x_2 + \theta_2^2 \exp(-\theta_1 - \theta_2)$
- Case B: $f(x, \theta) = x_1 x_2^2 + \cos(x_1 + x_2) + \exp(\theta_3 - x_1) + \frac{x_2}{1 + x_2 + \theta_3} + \theta_1 x_2 + \theta_2 x_1$

In the first example (case A), we randomly generate $n = 50$ data points, each with 5 variables from $[0, 1]$ using Latin hypercube design – x_1 and x_2 are considered



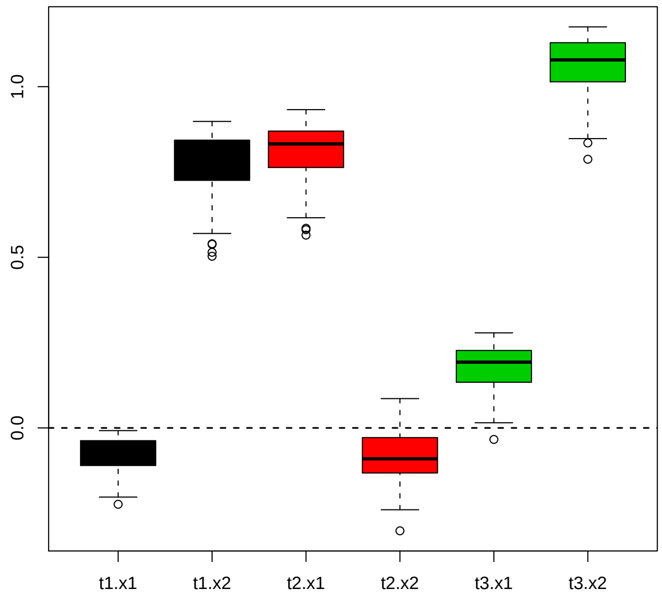


Figure 3.3: Reference distribution of $I_{jj'} - I_u$ for examples in section 3.5.2. The upper panel corresponds to case A, and the lower panel corresponds to case B.

as physical inputs and $\theta_1, \theta_2, \theta_3$ are considered as the calibration inputs. Note that

$$\frac{\partial^2 f(x,\theta)}{\partial\theta_1 \partial x_j} = \frac{\partial^2 f(x,\theta)}{\partial\theta_2 \partial x_j} = 0, \ j = 1, 2$$

Therefore both θ_1 and θ_2 are additive to the physical inputs x_j's, and according to Proposition 2, the calibration problem is non-identifiable. The second example (case B) has the same data generating mechanism as A. However, case B is identifiable.

We adopted the GaSP as the statistical emulator. Figure 3.3 shows the results with the distribution of $I_{jj'} - I_u$ over 100 replications. As we can see, in case A, our method correctly identifies θ_1 and θ_2 are additive with other physical inputs, while in case B, θ_1 is additive with x_1 and θ_2 is additive with x_2, while all other physical/calibration input pairs are non-additive.

3.6 Non-identifiability caused by “multiple solutions”

As we discussed at the beginning of the chapter, there are two types of non-identifiability in calibration: the first type is what we call “intrinsically non-identifiable”, and the other type is the so-called “multiple solutions”, where there exist multiple but finite number of calibration points which lead to the same function.

For example, consider $f(x_1, x_2, \theta_1, \theta_2, \theta_3) = 1 + \theta_1 x_1^2 + \theta_2 x_2 + (\theta_3 - 0.5)^2 x_3$, where $x_j, \theta_{j'} \in [0, 1]$, the corresponding calibration problem is not intrinsically non-identifiable, i.e, one can not find a smaller number of parameters to substitute $(\theta_1, \theta_2, \theta_3)$. However, we can also see that if $(\theta_1, \theta_2, \theta_3)$ are the true parameters, then $(\theta_1, \theta_2, 1 - \theta_3)$ is an equivalent truth. In such cases, we may consider marginal plots (or sensitivity plots) of y versus individual θ_1, θ_2 and θ_3 by averaging over the other dimensions; if a marginal plot is not monotonic over its support, it implies that the calibration

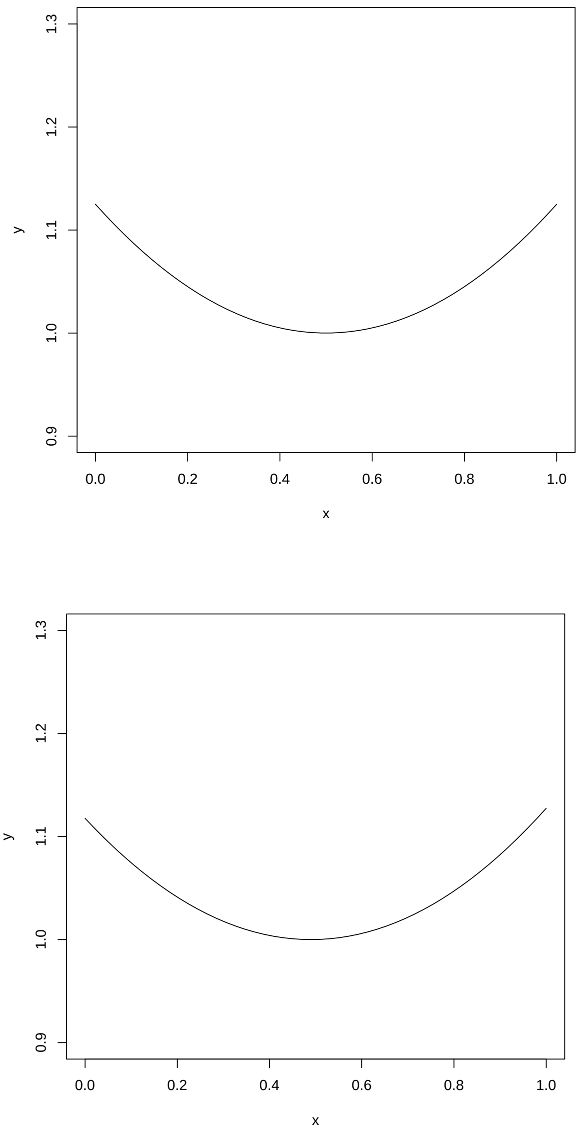


Figure 3.4: A non-identifiable problem caused by multiple solutions. The marginal plot shows y vs θ_3 is non-monotinic. The left panel shows the result using EMARS, and the right panel shows that using GaSP.

problem can be non-identifiable due to “multiple solutions”.

We use a simulation example to illustrate the point. Specifically, we consider:

$$f(x,\theta) = 1 + \theta_1 x_1^2 + \theta_2 x_2 + (\theta_3 - 0.5)^2 x_3 \quad (3.23)$$

$$Y^f(z) = f(z, \theta_1 = 0.5, \theta_2 = 0.5, \theta_3 = 0.2) + \epsilon \quad (3.24)$$

where $x_j, \theta_{j'} \in [0,1]$ and $\epsilon \sim N(0, 0.01)$. Note that although we set $\theta^* = (0.5, 0.5, 0.2)$, apparently there are two optimal solution, i.e $\hat{\theta^*} = (0.5, 0.5, 0.2)$ and $\hat{\theta^*} = (0.5, 0.5, 0.8)$.

For the computer data, we generate 100 data points in $[0,1]^5$ using the Latin hypercube design, and for the field data, we generate 10 data points in $[0,1]^2$ also using the Latin hypercube design. First, we use the method in section 3.4.3 to check whether the calibration problem is “intrinsically identifiable” and obtained $\hat{p} = 0.97$, which implies that the problem is not intrinsically non-identifiable. Next, we use both GaSP and EMARS to estimate $y^c(x,\theta)$ and obtain the marginal plots of $\hat{y}$ vs $\theta_{j'}$ (see Figure 3.4). As we can see, the relationship between $\hat{y}$ and θ_3 is non-monotonic, which implies that the calibration problem is non-identifiable due to “multiple solutions”.

3.7 A comparison study of Bayesian GaSP and EMARS in identifiable cases

In the above sections, we discussed how to detect possible non-identifiability in calibration. In this section, we focus on identifiable cases and compare EMARS with Bayesian GaSP for parameter calibration.

We consider 4 simulation examples, which are specified as follows:

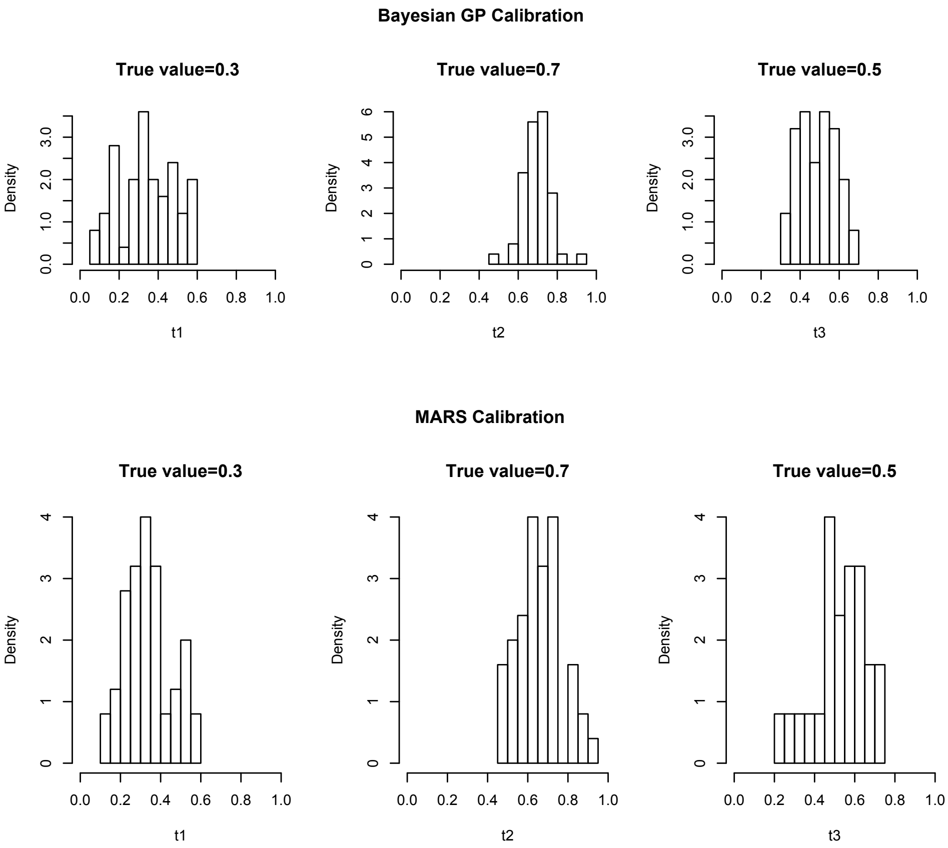


Figure 3.5: Distribution of the calibration estimates on case A of section 3.7 using Bayesian GaSP and EMARS.

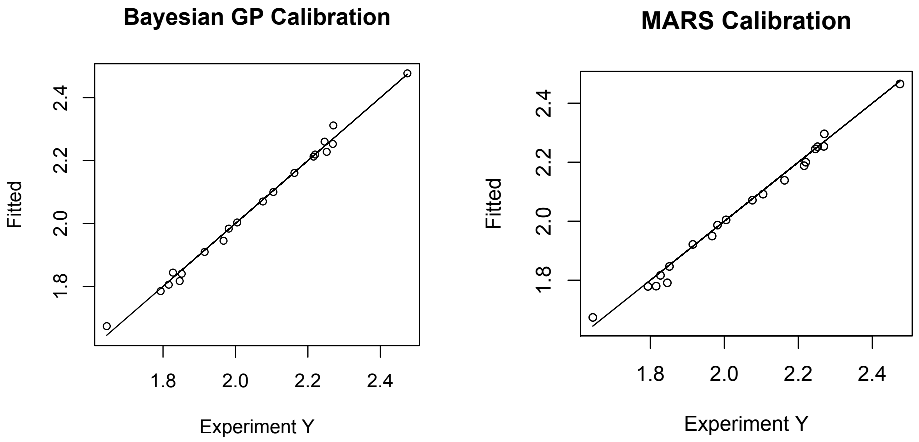


Figure 3.6: Prediction on an independent validation field data set using Bayesian GaSP and EMARS on case A of section 3.7.

- Case A: $f(x,\theta) = 0.2 + x_1 x_2^2 + \cos(x_1 + x_2) + 0.5 * \theta_3 \exp(-x_1) + 1.5 * \sin(\theta_3) x_2 + \theta_1 x_2 + \theta_2 x_1$
- Case B: $f(x,\theta) = 1 + e^{-\theta_1(x_1 + x_2^2)} + \frac{1}{1+\theta_2}(x_3^3 + x_4^2) + \sin(x_5 - \theta_3)$
- Case C: $f(x,\theta) = 1 + e^{-\theta_1 \sum_{j=1}^{10} x_j} + \frac{1}{1+\theta_2}(\sum_{j=11}^{14} x_j^2) + \sin(x_{15} - \theta_3)$
- Case D: $f(x,\theta) = 1 + e^{-\theta_1 \sum_{j=1}^{10} x_j} + \frac{1}{1+\theta_2}(\sum_{j=11}^{14} x_j^2) + \sin(x_{15} - \theta_3) + \frac{1}{100}(\sum_{j=16}^{25} x_j)$

Note that case A has 5 inputs, with x_1 and x_2 being physical inputs and θ_1, θ_2 and θ_3 calibration inputs. We generate $n = 50$ data points on $[0,1]^5$ using the Latin hypercube design. In addition to the 50 simulator data points, we also generate 10 field data points with calibration parameter $\theta_1 = 0.3$, $\theta_2 = 0.7$, $\theta_3 = 0.5$, i.e.

$$Y^f(z) = f(z, \theta = (0.3, 0.7, 0.5)) + \epsilon$$

where $\epsilon \sim N(0, \sigma^2)$ and $\sigma = 0.1 \times std(Y^c)$. To compare the prediction performance of different methods, we also generate an independent field data set with 20 data points so that one can compute the mean squared error $mse = \frac{1}{20}\sum_{i=1}^{20}(y_i^f - \hat{y}_i)^2$.

The dimensions in cases B, C, D are higher. Case B has 5 physical inputs and 3 calibration inputs; Case C has 15 physical inputs and 3 calibration inputs; Case D has 25 physical inputs and 3 calibration inputs. The sample sizes are chosen as: $n_B = 50, n_C = 100, n_D = 150$. Note that case C is a "sparse" case in the sense that inputs $x_{16}, \ldots, x_{25}$ do not have much impact on y^c. Everything else is the same as case A except that we fix $\theta_1^* = \theta_2^* = \theta_3^* = 0.5$ for all three cases when generating the field data.

We use both the GaSP method and the EMARS method to estimate the θ^* and further predict the field output y^f. For the Bayesian GaSP method, we use priors de-

Table 3.4: Comparison of calibration between GaSP and EMARS on simulation examples in section 3.7

Cases	Model	estimates of θ^* (sd)			$\sqrt{\text{mse}}$ of Y^f
		$\hat{\theta}_1^*$	$\hat{\theta}_2^*$	$\hat{\theta}_3^*$	on validation data
A	Bayesian GP	0.32(0.0215)	0.67(0.016)	0.47(0.024)	0.03
A	EMARS	0.29(0.0311)	0.69(0.010)	0.51(0.018)	0.02
B	Bayesian GP	0.47(0.024)	0.44(0.036)	0.49(0.028)	0.04
B	EMARS	0.51(0.021)	0.48(0.033)	0.50(0.019)	0.03
C	Bayesian GP	0.42(0.032)	0.57(0.041)	0.55(0.029)	0.07
C	EMARS	0.43(0.027)	0.55(0.033)	0.48(0.031)	0.04
D	Bayesian GP	0.40(0.038)	0.39(0.044)	0.58(0.035)	0.13
D	EMARS	0.45(0.031)	0.54(0.046)	0.54(0.028)	0.05

scribed in section 3.2 and 5000 MCMC samples for inference. For EMARS, we fit the model allowing three-way interactions, and the model is tuned using cross-validation. To make inference on θ, we use $B = 100$ bootstrap samples.

Figures 3.5 and 3.6 respectively show the calibration distribution and prediction on field data for case A, using both Bayesian GaSP and EMARS. Table 3.4 summarizes the calibration parameter estimates and MSE on the independent test set. As we can see, EMARS in general outperforms GaSP in terms of both parameter calibration and output prediction.

3.8 A case study – calibration with CRASH

In this study, the data comes from the Center of Radiative Shock-wave Hydrodynamics at the University of Michigan, where scientists conduct research on the shock-wave hydrodynamics caused by lasers with high energy and high velocity. The simulator data are based on computer code that depend on 8 inputs, 4 of which are considered calibration inputs, with a total size of 319 data points. There are also 9 field data points, which are collected by conducting the experiment in an Xenon filled tube, using laser to shock a Beryllium disk. The four regular inputs are Beryllium

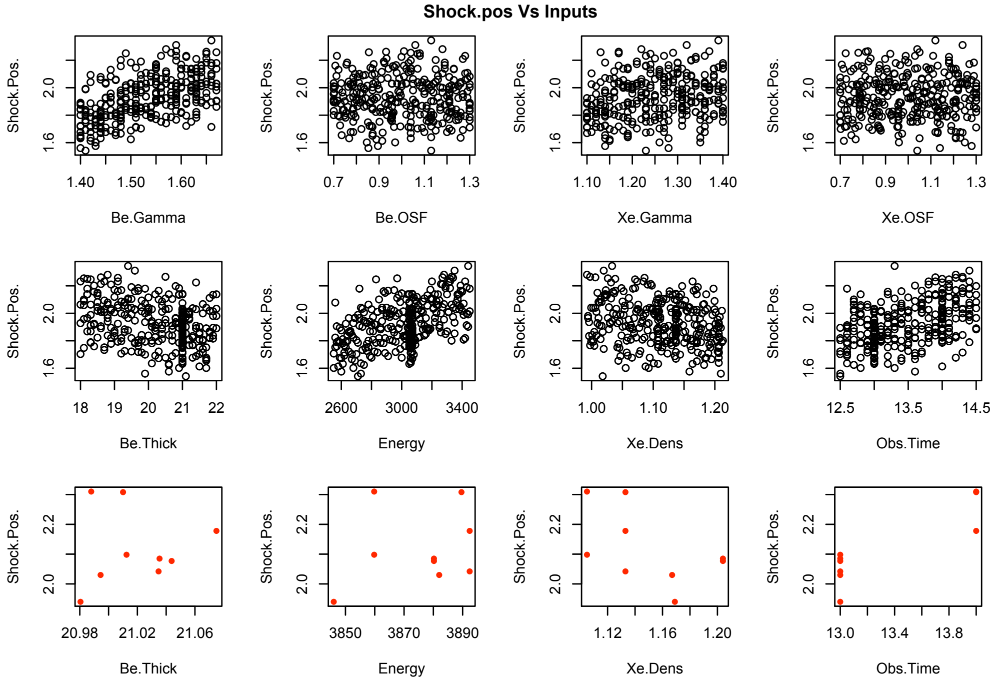


Figure 3.7: Scatter plot of both the simulator data and field data in CRASH example.

thickness, laser energy, Xenon density and observation time, while the four calibration inputs are Beryllium gamma, Xenon gamma, Beryllium.OSF (opacity scale factor) and Xenon.OSF. Figure 3.7 contains scatter plots of both the simulator data and field data. Figure 3.8 shows the calibration results using the Bayesian Gaussian Process method. As we can see, the distributions of the 4 calibration parameters scatter all around, which is not what scientists had expected. The leave-one-out prediction on the right panel of figure 3.8 also shows problems.

We suspected there might be a parameter identifiability issue with this problem, and applied the method in section 3.4.4. Since the computer code here is not “cheap”, we have to rely on an emulator to calculate the derivate matrix. We used EMARS in this case. Using the formulation (3.19) and (3.21), we calculated $\hat{p}$ and the results are shown in the upper half of table 3.5. Based on the values of $\hat{p}$, it is clear that this particular calibration problem is non-identifiable.

Table 3.5: Checking calibration identifiability in the CRASH study. The upper table shows the results of the original CRASH calibration problem, while the lower table shows the modified problem.

Reference choice	$\hat{C}$	$Avg(C^*)$	$\hat{p}$	Identifiable?
$\tilde{f} = f(x,\theta) + b(u_1 + u_2)$	2.5E-7	3.2E-8	7%	No
$\tilde{f} = f(x,\theta) + b(u_1 u_2)$	2.5E-7	4.3E-8	6%	No
$\tilde{f} = f(x,\theta) + b(u_1 + u_2)$	1.5E-2	4.8E-8	96%	Yes
$\tilde{f} = f(x,\theta) + b(u_1 u_2)$	1.5E-2	7.5E-8	96%	Yes

Further, we also applied the special additivity condition in Proposition 2, and Figure 3.9 shows the result. Based on the distributions of the 4×4 referenced $I_{jj'}$, it is clear that only 3 interaction terms are significant – Beryllium thickness versus Beryllium gamma, Laser energy versus Beryllium gamma, and Laser energy versus Xenon gamma. This implies that the other two calibration inputs (the opacity scale parameters) are additive to all the physical inputs, which strengthens the conclusion that the calibration problem here is non-idenfiable. Further analysis by EMARS shows that these two calibration inputs do not contribute much to the fitted model of EMARS. After excluding these two calibration inputs which do not have much effects on the response, we redid the analysis and Figure 3.10 shows the new calibration distributions and the leave-one-out prediction of experimental response. The lower half of Table 3.5 now confirms that the modified CRASH calibration problem is indeed identifiable. Further, the Beryllium gamma concentrates near 1.47 and Xenon gamma concentrates near 1.27, which are more interpretable and the leave-one-out prediction accuracy is also improved.

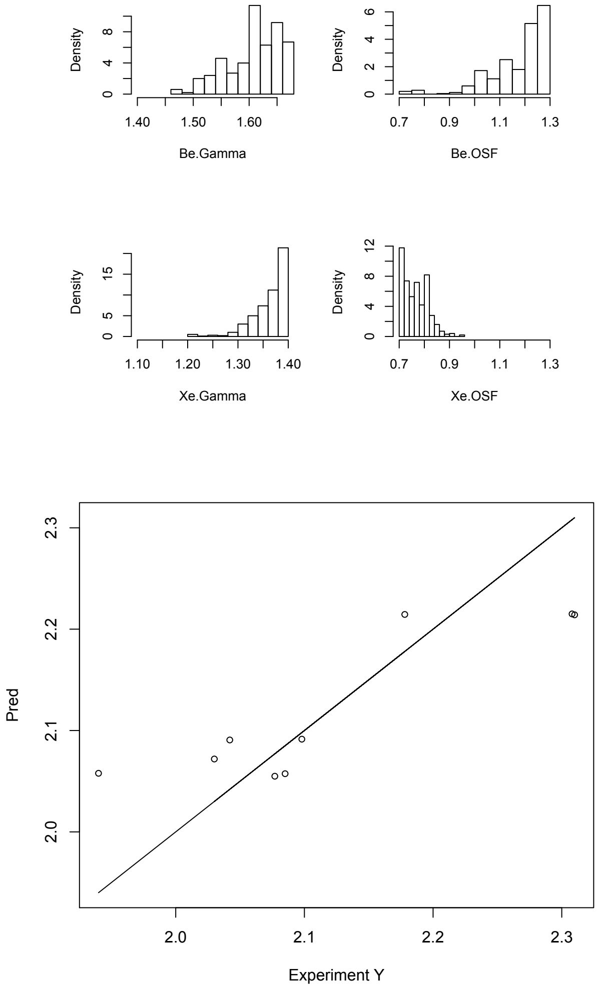


Figure 3.8: Bayesian Gaussian Process calibration on 4 calibration parameters and leave-one-out predictions on the field response.

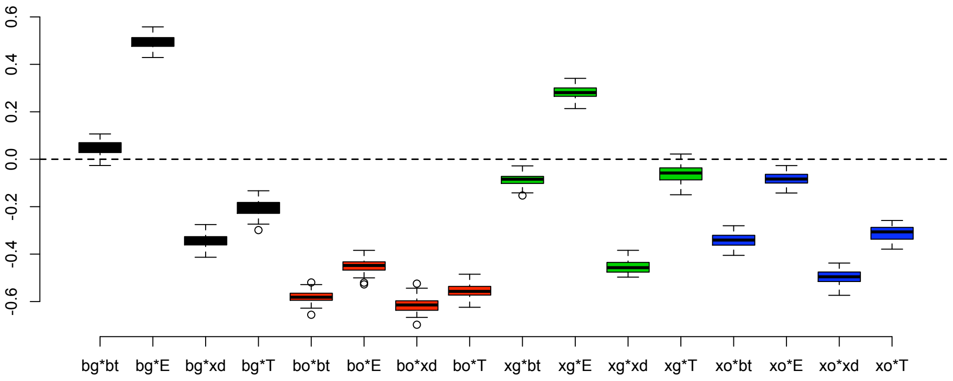


Figure 3.9: Reference $I_{jj'-I_u}$ over 4×4 interactions in the case study.

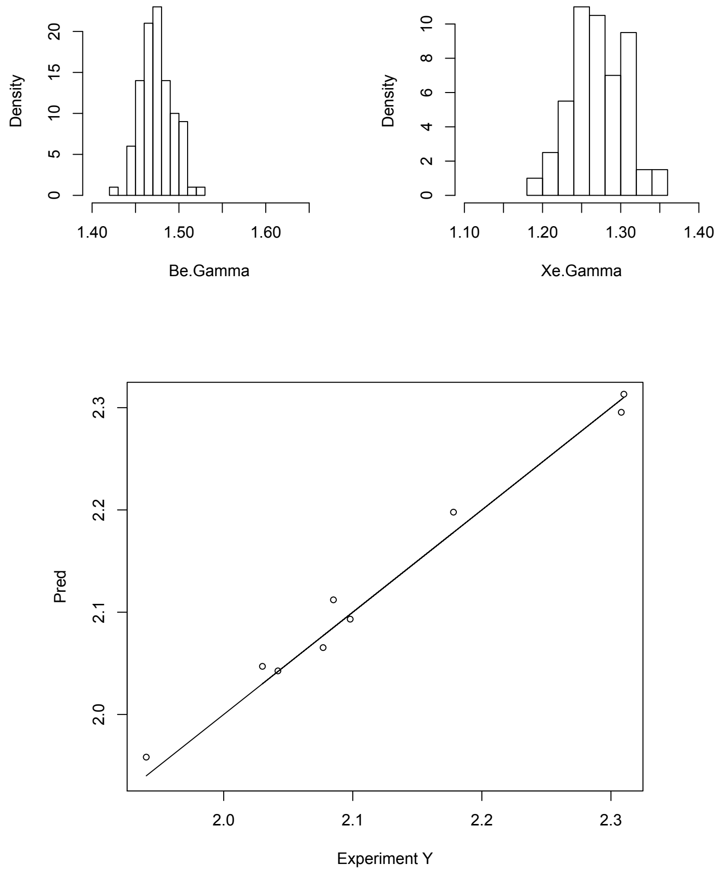


Figure 3.10: Distribution of the estimated calibration parameter (using EMARS) considering only two identifiable calibration inputs and leave-one-out predictions on the field response.

3.9 Summary

In this chapter, we have focused on the potential non-identifiability issue in the computer model calibration problem. We defined two types of non-identifiability, the “intrinsically non-identifiable”, and the non-identifiability caused by “multiple solutions”. We offered sufficient and necessary conditions to check the intrinsic non-identifiability. We also proposed numerical methods to implement these conditions. Numerical studies indicated that the proposed method works well, and the case study provided meaningful and promising results.

CHAPTER IV

Future Studies

In this thesis, we have focused on two aspects in the modeling of computer experiments: one is the accuracy of statistical emulators in predicting computer experiments, and the other is the calibration problem.

On the first topic, we compared the predictive accuracy of the EMARS approach and others versus Bayesian GaSP and found that the EMARS approach is a good alternative under a variety of situations. There are, however, a number of points that need further study. There are two different versions of EMARS that were discussed in Chapter 2. Interestingly, Version Two performed slightly better than Version One, but the latter is more natural as it includes polynomials in $[x-v]_+$ and $[v-x]_+$. The reason for the better performance of Version Two needs further investigation. In addition, the comparisons are all made under Latin hypercube designs, which have been widely used in the literature. In future studies, one interesting question is whether the performance of different emulators depends on different types of design. The other related question is for a specific emulator, what will be its optimal design given a fixed number of design points? For instance, what will be the optimal design for an EMARS approach? Another question is comparing the performance of the emulators under other criteria such as predicting the maximum or minimum of a function. Finally, a very interesting question is the development of uncertainty regions for the regression-based methods since the usual assumptions of random and iid errors are not valid. We have tried to use bootstrap methods but this question needs to be further studied.

On the second topic of the calibration, we discussed the potential identifiability issues and developed several statistical methods to detect such issues. However, it should be noted that proposition 1 in chapter 3 discussed the identifiability condition in a global fashion, where the conditions listed need to be satisfied for all $\theta \in \Theta$. There are cases where calibration parameters are locally identifiable. Consider the following continuous 3-dimensional function

$$f(x, \theta_1, \theta_2) = x(\theta_1 + \theta_2) + [\theta_2 - 0.5]_+$$

where $x, \theta_1, \theta_2 \in [0, 1]^3$. Clearly we see that when $\theta_2 <= 0.5$, there is no way to differentiate θ_1 and θ_2. But when $\theta_2 > 0.5$, this is an identifiable case. More research is needed to study this problem.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to mcmc for machine learning. *Machine learning*, 50(1-2):5–43, 2003.

[2] RA Bates, RJ Buck, E Riccomagno, and HP Wynn. Experimental design and observation for large systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 77–94, 1996.

[3] MJ Bayarri, JO Berger, J Cafeo, G Garcia-Donato, F Liu, J Palomo, RJ Parthasarathy, R Paulo, J Sacks, and D Walsh. Computer model validation with functional output. *The Annals of Statistics*, pages 1874–1906, 2007.

[4] E Ben-Ari and D Steinberg. Modeling data from computer experiments: An empirical comparison of kriging with mars and projection pursuit regression. *Quality Engineering*, 19:4:327–338, 1994.

[5] Alain Berlinet and Christine Thomas-Agnan. *Reproducing kernel Hilbert spaces in probability and statistics*, volume 3. Kluwer Academic Boston, 2004.

[6] Scott M Berry, Raymond J Carroll, and David Ruppert. Bayesian smoothing and regression splines for measurement error problems. *Journal of the American Statistical Association*, 97(457):160–169, 2002.

[7] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[8] Peter Bühlmann and Bin Yu. Boosting with the l 2 loss: regression and classification. *Journal of the American Statistical Association*, 98(462):324–339, 2003.

[9] E Catchpole and B J.T. Morgan. Detecting parameter redundancy. *Biometrika*, 84:1:187–196, 1997.

[10] P S Craig, M Goldstein, J C Rougier, and A H Seheult. Bayesian forecasting using large computer models. *Journal of the American Statistical Association*, 96:717–729, 2001.

[11] P Craven and G Wahba. Smoothing noisy data with spline functions: Estimating the correct degree of smoothing by the method of generalized cross-validation. *Numerische Mathematik*, 31:317–403, 1979.

[12] Carla Currin, Toby Mitchell, Max Morris, and Don Ylvisaker. Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments. *Journal of the American Statistical Association*, 86(416): 953–963, 1991.

[13] Sarat C Dass and Vijayan N Nair. Edge detection, spatial smoothing, and image reconstruction with partially observed multivariate data. *Journal of the American Statistical Association*, 98(461):77–89, 2003.

[14] C DeBoor. *A Practical Guide to Splines*. Springer-Verlag, New York, 1978.

[15] Thomas G Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine learning*, 40(2):139–157, 2000.

[16] B Efron, T Hastie, I Johnstone, and R Tibshirani. Least angle regression. *The Annals of Statistics*, 32:407–499, 2004.

[17] Y Freund and R Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1): 119–139, 1997.

[18] Y Freund and R Schapire. Improved boosting algorithms using condence-rated predictors. *Machine Learning*, 37(3):297–336, 1999.

[19] J Friedman. Multivariate adaptive regression splines (with discussion). *Annals of Statistics*, 19:1–141, 1991.

[20] J Friedman. Greedy function approximation: a gradient boosting machine. *Technical Report , Dept. of Statistics, Stanford University*, 1999.

[21] J Friedman, T Hastie, and R Tibshirani. Additive logistic regression: a statistical view of boosting. *Technical Report*, 1998.

[22] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407, 2000.

[23] Jerome H Friedman. Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378, 2002.

[24] David E Goldberg and John H Holland. Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99, 1988.

[25] R Gramacy and H Lee. Bayesian treed gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103(483):1119–1130, 2008.

[26] R Gramacy and H Lee. Bayesian treed gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103(483):1119–1130, 2008.

[27] P J Green and B W Silverman. *Nonparametric Regression and Generalized Linear Models*. Chapman and Hall, London, 1994.

[28] Peter J Green. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

[29] C Gu. *Smoothing Spline ANOVA Models*. Springer series in statistics, New York, 2002.

[30] Gang Han, Thomas J Santner, and Jeremy J Rawlinson. Simultaneous determination of tuning and calibration parameters for computer experiments. *Technometrics*, 51(4):464–474, 2009.

[31] T J Hastie and R J Tibshirani. *Generalized Additive Models*. Chapman and Hall, London, 1990.

[32] Trevor Hastie, Robert Tibshirani, Jerome Friedman, and James Franklin. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, 27(2):83–85, 2005.

[33] Jon C Helton and Freddie Joe Davis. Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems. *Reliability Engineering & System Safety*, 81(1):23–69, 2003.

[34] D Higdon, M Kennedy, J Cavendish, J Cafeo, and R Ryne. Combining field data and computer simulations for calibration and prediction. *SIAM J. Sci. Comput.*, 26:448–466, 2004.

[35] D Higdon, J Gattiker, B Williams, and M Rightley. Computer model calibration using high-dimensional output. *Journal of the American Statistical Association*, 103(482):570–583, 2008.

[36] J Holloway, D Bingham, C Chou, P Drake, B Fryxell, M Grosskopf, B Holst, B Mallick, M Ryan, A Mukherjee, V Nair, K Powell, D Ryu, I Sokolov, G Toth, and Z Zhang. Predictive modeling of a radioative shock system. *Reliability Engineering and System Safety*, 10:1016, 2011.

[37] M Kennedy and A O'Hagan. Bayesian calibration of computer codels (with discussion). *Journal of the Royal Statistical Society, Series B*, 63:425–464, 2001.

[38] Marc C Kennedy and Anthony O'Hagan. Predicting the output from a complex computer code when fast approximations are available. *Biometrika*, 87(1):1–13, 2000.

[39] G S Kimeldorf and G Wahba. Some results on tchebychan spline functions. *Journal of Mathematical Analysis and Applications*, 33:82–95, 1970.

[40] JT Larsen and SM Lane. A plasma hydrodynamics code for dense plasma studies. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 51:179, 1994.

[41] Neil D Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. *Advances in neural information processing systems*, 16 (329-336):3, 2004.

[42] R Li and A Sudjianto. Analysis of computer experiments using penalized likelihood in gaussian kriging models. *Technometrics*, 74(2):111–120, 2005.

[43] Xihong Lin and Daowen Zhang. Inference in generalized additive mixed modelsby using smoothing splines. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(2):381–400, 1999.

[44] Y Lin and H Zhang. Component selection and smoothing in multivariate nonparametric regression. *The Annals of Statistics*, 34:2272–2297, 2006.

[45] D Liu, X Lin, and D Ghosh. Semiparametric regression for multi-dimensional genomic pathway data: Least square kernel machines and linear mixed modles. *Biometrics*, 63:1079–1088, 2007.

[46] Ryan G McClarren, D Ryu, R Paul Drake, Michael Grosskopf, Derek Bingham, Chuan-Chih Chou, Bruce Fryxell, Bart Van der Holst, James Paul Holloway, Carolyn C Kuranz, et al. A physics informed emulator for laser-driven radiating shock simulations. *Reliability Engineering & System Safety*, 96(9):1194–1207, 2011.

[47] T J Mitchell, M D Morris, and D Ylvisaker. Two-level fractional factorials and bayesian prediction. *Statistica Sinica*, 5:559–573, 1995.

[48] Max D Morris and Toby J Mitchell. Exploratory designs for computational experiments. *Journal of statistical planning and inference*, 43(3):381–402, 1995.

[49] Max D Morris, Toby J Mitchell, and Donald Ylvisaker. Bayesian design and analysis of computer experiments: use of derivatives in surface prediction. *Technometrics*, 35(3):243–255, 1993.

[50] Vijay Nair, Mark Hansen, and Jan Shi. Statistics in advanced manufacturing. *Journal of the American Statistical Association*, 95(451):1002–1005, 2000.

[51] Vijay N Nair, Luis A Escobar, and Michael S Hamada. Design and analysis of experiments for reliability assessment and improvement. In *Mathematical Reliability: An Expository Perspective*, pages 161–182. Springer, 2004.

[52] J Oakley and A OHagan. Bayesian inference for the uncertainty distribution of computer model outputs. *Biometrika*, 89(4):769–784, 2002.

[53] Margaret A Oliver and R Webster. Kriging: a method of interpolation for geo-

graphical information systems. *International Journal of Geographical Information System*, 4(3):313–332, 1990.

[54] Peter ZG Qian, Huaiqing Wu, and CF Jeff Wu. Gaussian process models for computer experiments with qualitative and quantitative factors. *Technometrics*, 50(3), 2008.

[55] Adrian E Raftery and Steven M Lewis. Implementing mcmc. *Markov chain Monte Carlo in practice*, pages 115–130, 1996.

[56] C Rasmussen and C Williams. *Gaussian Process for Machine Learning.* MIT Press, 2006.

[57] John Rice and Murray Rosenblatt. Smoothing splines: regression, derivatives and deconvolution. *The annals of Statistics*, 11(1):141–156, 1983.

[58] Herbert Robbins. Some aspects of the sequential design of experiments. In *Herbert Robbins Selected Papers*, pages 169–177. Springer, 1985.

[59] Saharon Rosset, Ji Zhu, and Trevor Hastie. Boosting as a regularized path to a maximum margin classifier. *The Journal of Machine Learning Research*, 5: 941–973, 2004.

[60] J Sacks, W J Welch, T J Mitchell, and H P Wynn. Design and analysis of computer experiments. *Statistical Science*, 4:409–435, 1989.

[61] T Santner, J Williams, and W Notz. *The Design and Analysis of Computer Experiments.* Springer series in statistics, New York, 2005.

[62] B Scholkoph and A Smola. *Learning with Kernels.* MIT Press, 2002.

[63] W Silverman. Some aspects of the spline smoothing approach to non-parametric regression curve fitting. *Journal of the Royal Statistical Society, Series B*, 47: 1–52, 1985.

[64] Adrian FM Smith and Gareth O Roberts. Bayesian computation via the gibbs sampler and related markov chain monte carlo methods. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 3–23, 1993.

[65] Daniel Sorensen and Daniel Gianola. *Likelihood, Bayesian and MCMC methods in quantitative genetics.* Springer, 2002.

[66] T Speed. Discussion to blup is a good thing: The estimation of random eects by robinson, g. k. *Statistical Science*, 6:42–44, 1991.

[67] M Stein. *Interpolation of Spatial Data: Some Theory for Kriging.* Springer-Verlag, New York, 1999.

[68] Michael Stein. Large sample properties of simulations using latin hypercube sampling. *Technometrics*, 29(2):143–151, 1987.

[69] Michael Leonard Stein. *Interpolation of spatial data: some theory for kriging.* Springer, 1999.

[70] Max J Suarez, MM Rienecker, R Todling, J Bacmeister, L Takacs, HC Liu, W Gu, M Sienkiewicz, RD Koster, R Gelaro, et al. The geos-5 data assimilation system-documentation of versions 5.0. 1, 5.1. 0, and 5.2. 0. 2008.

[71] R Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1996.

[72] V Vapnik. *Statistical Learning Theory.* Wiley, New York, 1998.

[73] Arũnas P Verbyla, Brian R Cullis, Michael G Kenward, and Sue J Welham. The analysis of designed experiments and longitudinal data by using smoothing splines. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 48(3):269–311, 1999.

[74] G Wahba. *Spline Models for Observational Data*. SIAM, Philadelphia, 1990.

[75] G Wahba. *Support Vector Machines, Reproducing Kernel Hilbert Spaces and The Randomized GACV*. Technical Report, Department of Statistics, University of Wisconsin, Madison, 1997.

[76] Grace Wahba. Spline interpolation and smoothing on the sphere. *SIAM Journal on Scientific and Statistical Computing*, 2(1):5–16, 1981.

[77] William J Welch, Robert J Buck, Jerome Sacks, Henry P Wynn, Toby J Mitchell, and Max D Morris. Screening, predicting, and computer experiments. *Technometrics*, 34(1):15–25, 1992.

[78] CF Jeff Wu and Michael S Hamada. *Experiments: planning, analysis, and optimization*, volume 552. John Wiley & Sons, 2011.

[79] Dong Xiang and Grace Wahba. A generalized approximate cross validation for smoothing splines with non-gaussian data. *Statistica Sinica*, 6:675–692, 1996.

[80] Ji Zhu and Trevor Hastie. Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics*, 14(1), 2005.